What is TorqueScript

TorqueScript (TS) is a proprietary scripting language developed specifically for Torque technology. The language itself is derived from the scripting used for Tribes 2, which was the base tech Torque evolved from. Scripts are written and stored in .cs files, which are compiled and executed by a binary compiled via the C++ engine (.exe for Windows or .app OS X).

The CS extension stands for "C Script," meaning the language resembles C programming. Though there is a connection, TorqueScript is a much higher level language and is easier to learn than standard C or C++. 
Basic Usage

Like most other scripting languages, such as Python or Java Script, TorqueScript is a high-level programming language interpreted by Torque 3D at run time. Unlike C++, you can write your code in script and run it without recompiling your game.

All of your interfaces can be built using the GUI Editor, which saves the data out to TorqueScript. The same goes for data saved by the World Editor or Material Editor. Most of the editors themselves are C++ components exposed and constructed via TorqueScript.

More importantly, nearly all of your game play programming will be written in TorqueScript: inventory systems, win/lose scenarios, AI, weapon functionality, collision response, and game flow. All of these can be written in TorqueScript. The language will allow you to rapidly prototype your game without having to be a programming expert or perform lengthy engine recompilation.

Copyright © GarageGames, LLC. All Rights Reserved.
Scripting vs Engine Programming

As mentioned above, TorqueScript is comprised of the core C++ objects needed to make your game. For example, you will use the PlayerData structure to create player objects for your game. This structure was written in C++:

**C++ PlayerData Code**

```c++
struct PlayerData: public ShapeBaseData {
    typedef ShapeBaseData Parent;

    bool renderFirstPerson; // Render the player
    mass = 9.0f; // from ShapeBase
    drag = 0.3f; // from ShapeBase
    density = 1.1f; // from ShapeBase
}
```

Instead of having to go into C++ and create new PlayerData objects or edit certain fields (such as mass), PlayerData was exposed to TorqueScript:

**Example TorqueScript PlayerData Code**

```text
 datablock PlayerData(DefaultPlayerData) {
    renderFirstPerson = true;
    className = Armor;
    shapeFile = "art/shapes/actors/gideon/base"
    mass = 100;
    drag = 1.3;
    maxdrag = 0.4;
}
```
// Allowable Inventory Items
maxInv[Pistol] = 1;
maxInv[PistolAmmo] = 50;
};

If you want to change the name of the object, the mass, the inventory, or anything else, just open the script, make the change, and save the file. When you run your game, the changes immediately take affect. Of course, for this example you could have used the Datablock Editor, but you should get the point. TorqueScript is the first place you should go to write your game play code.

Copyright © GarageGames, LLC. All Rights Reserved.
Script Editors

TorqueScript files are essentially text files. This means you have several editors to choose from. Some users prefer to use the stock OS text editors: Notepad on Windows or Text Edit on OS X. Others will use their programming IDEs (Interactive Development Environments), such as Visual Studio for Windows or Xcode on OS X. Third party applications are the most popular choice:

On Windows

Recommended:

- **Torsion** - Torsion is undeniably the best TorqueScript IDE. It was developed by Torque veterans Sickhead Games. If you are developing on Windows, this is the first thing you should purchase after Torque 3D. No other editor offers this level of quality and functionality:
  - Integrated "One Click" script debugging
  - Full control over script execution via step and break commands
  - Advanced editor features like code folding, line wrapping, auto-indent, column marker, automatic bracket matching, and visible display of tabs and spaces
  - Goto line and text searching
  - ScriptSense updated dynamically as you type
  - Customizable syntax highlighting for TorqueScript
  - Unlimited undo/redo buffer
  - Code browser window for exploring both engine exports and script symbols in your project

Alternatives:

- **Notepad++** - This is a free (as in "free speech" and also as in "free beer") source code editor and Notepad replacement that supports several languages.
• **UltraEdit** - UltraEdit is a powerful disk-based text editor, programmer's editor, and hex editor that is used to edit TorqueScript, HTML, PHP, javascript, Perl, C/C++, and a multitude of other coding/programming languages.

**On OS X**

**Recommended:**

• **Xcode** - Xcode is Apple's premiere development environment for Mac OS X. If you plan on modifying Torque 3D's source code, you will need this anyway. Most developers at GarageGames use Xcode to modify their scripts on a Mac.

• **Text Edit** - This is the OS X default text editor. With no bells or whistles, this is not the best editor you can use on OS X, but it is free and ships with the OS.

**Alternatives:**

• **TIDE** - Torque Integrated Development Environment (TIDE) is a free, cross-platform IDE for Torque Game Engine scripting by Paul Dana and Stefan Moises. It is implemented in Java as a "plugin suite" for the jEdit text editor and contains plugins for syntax highlighting, function browsing, script debugging, etc.

• **Smultron** - Smultron is a text editor written in Cocoa for Mac OS X Leopard 10.5 which is designed to be both easy to use and powerful.

Copyright © GarageGames, LLC. All Rights Reserved.
Recent Changes

Foreach Statements

Two new statements simplify the iteration over sets of objects and string vectors.

To loop over each object in a SimSet, use the foreach statement:

```plaintext
foreach( %obj in %set )
  /* do something with %obj */;
```

To loop over each element in a string vector, use the foreach$ statement:

```plaintext
foreach$( %str in "a b c" )
  /* do something with %str */;
```

foreach Statement foreach$ Statement
Redefinition Behavior
Singletons
Lookup Operator
Floating-Point Notation
Datablock Syntax Extensions
Syntax Guide

- Syntax Guide
  - The Basics
  - Variables
  - Data Types
  - Operations
  - Control Statements
  - Loops
  - Functions
  - Game Objects
  - Packages

Copyright © GarageGames, LLC. All Rights Reserved.
The Basics

Rules

Like other languages, TorqueScript has certain syntactical rules you need to follow. The language is typeless, very forgiving, easy to debug, and is not as strict as a low level language like C++. Observe the following line in a script:

```plaintext
// Create test variable with a temporary variable
%testVariable = 3;
```

The three most simple rules obeyed in the above code are:

1. Ending a line with a semi-colon (;)
2. Proper use of white space.
3. Commenting

The engine will parse code line by line, stopping whenever it reaches a semi-colon. This is referred to as a statement termination, common to other programming languages such as C++, Javascript, etc. The following code will produce an error that may cause your entire script to fail:

To the human eye, you are able to discern two separate lines of code with different actions. Here is how the script compiler will read it:

```
%testVariable = 3%anotherVariable = 4;
```

This is obviously not what the original code was meant to do. There are exemptions to this rule, but they come into play when multiple lines of code are supposed to work together for a single action:

```
if(%testVariable == 4)  
  echo("Variable equals 4");
```
We have not covered conditional operators or echo commands yet, but you should notice that the first line does not have a semi-colon. The easiest explanation is that the code is telling the compiler: "Read the first line, do the second line if we meet the requirements." In other words, perform operations between semi-colons. Complex operations require multiple lines of code working together.

The second rule, proper use of whitespace, is just as easy to remember. Whitespace refers to how your script code is separated between operations. Let's look at the first example again:

```plaintext
%testVariable = 3;
```

The code is storing a value (3) in a local variable (testVariable). It is doing so by using a common mathematical operator, the equal sign. TorqueScript recognizes the equal sign and performs the action just as expected. It does not care if there are spaces in the operation:

```plaintext
%testVariable=3;
```

The above code works just as well, even without the spaces between the variable, the equal sign, and the 3. The whitespace rule makes a lot more sense when combined with the semi-colon rule and multiple lines of code working together. The following will compile and run without error:

```plaintext
if(%testVariable == 4) echo("Variable equals
```
Comments

The last rule is optional, but should be used as often as possible if you want to create clean code. Whenever you write code, you should try to use comments. Comments are a way for you to leave notes in code which are not compiled into the game. The compiler will essentially skip over these lines.

There are two different comment syntax styles. The first one uses the two slashes, //. This is used for single line comments:

Example:

```
// This comment line will be ignored
// This second line will also be ignored
%testVariable = 3;
// This third line will also be ignored
```

In the last example, the only line of code that will be executed has to do with testVariable. If you need to comment large chunks of code, or leave a very detailed message, you can use the /* comment */ syntax (without spaces between asterix and slash). The /* starts the commenting, the */ ends the commenting, and anything in between will be considered a comment.

Example:

```
/*
While attending school, an instructor taught
"Read. Read Code. Code."
Applying this to Torque 3D development is eas
READ the documentation first.
```
READ CODE written by other Torque developers.

CODE your own prototypes based on what you have

As you can see, the comment makes full use of whitespace and multiple lines. While it is important to comment what the code does, you can also use this to temporarily remove unwanted code until a better solution is found:

Example:

```c
// Why are you using multiple if statements.
/*
  if(%testVariable == "Mich")
    echo("User name: ", %testVariable);

  if(%testVariable == "Heather")
    echo("User Name: ", %testVariable);

  if(%testVariable == "Nikki")
    echo("User Name: ", %testVariable);
*/
```

Copyright © GarageGames, LLC. All Rights Reserved.
Variables

Now that you know the two most basic rules of writing code in TorqueScript, this is the best time to learn about variables. A variable is a letter, word, or phrase linked to a value stored in your game's memory and used during operations. Creating a variable is a one line process. The following code creates a variable by naming it and assigning a value:

```torque
%localVariable = 3;
```

You can assign any type value to the variable you want. This is referred to as a language being **type-insensitive**. TorqueScript does not care (insensitive) what you put in a variable, even after you have created it. The following code is completely valid:

```torque
%localVariable = 27;
%localVariable = "Heather";
%localVariable = "7 7 7";
```

The main purpose of the code is to show that TorqueScript treats all data types the same way. It will interpret and convert the values internally, so you do not have to worry about typecasting. That may seem a little confusing. After all, when would you want a variable that can store a number, a string, or a vector?

You will rarely need to, which is why you want to start practicing good programming habits. An important practice is proper variable naming. The following code will make a lot more sense, considering how the variables are named:

```torque
%userName = "Heather";
%userAge = 27;
%userScores = "7 7 7";
```
Earlier, I mentioned that TorqueScript is more forgiving than low level programming languages. While it expects you to obey the basic syntax rules, it will allow you to get away with small mistakes or inconsistency. The best example is variable case sensitivity. At some point in school you learned the difference between upper case and lower case letters.

With variables, TorqueScript is not case sensitive. You can create a variable and refer to it during operations without adhering to case rules:

```torquescript
%userName = "Heather";
echo(%Username);
```

In the above code, userName and Username are the same variable, even though they are using different capitalization. You should still try to remain consistent in your variable naming and usage, but you will not be punished if you slip up occasionally.
Types

There are two types of variables you can declare and use in TorqueScript: local and global. Both are created and referenced similarly:

```torque
%localVariable = 1;
$globalVariable = 2;
```

As you can see, local variable names are preceded by the percent sign (%). Global variables are preceded by the dollar sign ($). Both types can be used in the same manner: operations, functions, equations, etc. The main difference has to do with how they are scoped.

In programming, scoping refers to where in memory a variable exists and its life. A local variable is meant to only exist in specific blocks of code, and its value is discarded when you leave that block. Global variables are meant to exist and hold their value during your entire programs execution. Look at the following code to see an example of a local variable:

```torque
function test()
{
    %userName = "Heather";
    echo(%userName);
}
```

We will cover functions a little later, but you should know that functions are blocks of code that only execute when you call them by name. This means the variable, userName, does not exist until the test() function is called. When the function has finished all of its logic, the userName variable will no longer exist. If you were to try to access the userName variable outside of the function, you will get nothing.
Most variables you will work with are local, but you will eventually want a variables that last for your entire game. These are extremely important values used throughout the project. This is when global variables become useful. For the most part, you can declare global variables whenever you want:

```php
$PlayerName = "Heather";

function printPlayerName()
{
    echo($PlayerName);
}

function setPlayerName()
{
    $PlayerName = "Nikki";
}
```

The above code makes full use of a global variable that holds a player's name. The first declaration of the variable happens outside of the functions, written anywhere in your script. Because it is global, you can reference it in other locations, including separate script files. Once declared, your game will hold on to the variable until shutdown.
Data Types

As I mentioned above, TorqueScript comprised of the core C++ objects needed to make your game. For example, you will use the PlayerData structure to create player objects for your game. This structure was written in C++:

TorqueScript implicitly supports several variable data-types: numbers, strings, booleans, and arrays and vectors. If you wish to test the various data types, you can use the echo(...) command. For example:

```plaintext
%meaningOfLife = 42;
echo(%meaningOfLife);
$name = "Heather";
echo($name);
```

The echo will post the results in the console, which can be accessed by pressing the tilde key (~) while in game.
# Numbers

TorqueScript handles standard numeric types

<table>
<thead>
<tr>
<th>Number</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>(Integer)</td>
</tr>
<tr>
<td>1.234</td>
<td>(floating point)</td>
</tr>
<tr>
<td>1234e-3</td>
<td>(scientific notation)</td>
</tr>
<tr>
<td>0xc001</td>
<td>(hexadecimal)</td>
</tr>
</tbody>
</table>

**Example:**

```torque
%myInteger = 3;
%myFloat = 2.5;
%mySciNot = 314e-1;
%myHexa = 0xc001;
```
**Strings**

Text, such as names or phrases, are supported as strings. Numbers can also be stored in string format. Standard strings are stored in double-quotes.

"abcd" (string)

Example:

$UserName = "Heather";
Tagged Strings

Strings with single quotes are called "tagged strings".

'abcd' (tagged string)

Tagged strings are special in that they contain string data, but also have a special numeric tag associated with them. Tagged strings are used for sending string data across a network. The value of a tagged string is only sent once, regardless of how many times you actually do the sending.

On subsequent sends, only the tag value is sent. Tagged values must be de-tagged before printing. You will not need to use a tagged string often unless you are in need of sending strings across a network often, like a chat system.

Example:

```
$a = 'This is a tagged string';
echo(" Tagged string: ", $a);
echo("Detagged string: ", detag('$a'));
```

The output will be similar to this:

```
24
```

The second echo will be blank unless the string has been passed to you over a network.
Booleans

Like most programming languages, TorqueScript also supports Booleans. Boolean numbers have only two values- true or false.

<table>
<thead>
<tr>
<th>true</th>
<th>(1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>(0)</td>
</tr>
</tbody>
</table>

Again, as in many programming languages the constant "true" evaluates to the number 1 in TorqueScript, and the constant "false" evaluates to the number 0. However, non-zero values are also considered true. Think of booleans as "on/off" switches, often used in conditional statements.

Example:

```php
$lightsOn = true;

if($lightsOn)
    echo("Lights are turned on");
```
Arrays

Arrays are data structures used to store consecutive values of the same data type.

$TestArray[n]$ (Single-dimension)
$TestArray[m,n]$ (Multidimensional)
$TestArray[m\_n]$ (Multidimensional)

If you have a list of similar variables you wish to store together, try using an array to save time and create cleaner code. The syntax displayed above uses the letters 'n' and 'm' to represent where you will input the number of elements in an array. The following example shows code that could benefit from an array:

**Example:**

$firstUser = "Heather";$
$secondUser = "Nikki";$
$thirdUser = "Mich";$

echo($firstUser);
echo($secondUser);
echo($thirdUser);

Instead of using a global variable for each user name, we can put those values into a single array:

**Example:**

$userNames[0] = "Heather";$
$userNames[1] = "Nikki";$

echo($userNames[0]);
echo($userNames[1]);
echo($userNames[2]);

Now, let's break the code down. Like any other variable declaration, you can create an array by giving it a name and value:

$userNames[0] = "Heather";

What separates an array declaration from a standard variable is the use of brackets[]. The number you put between the brackets is called the index. The index will access a specific element in an array, allowing you to view or manipulate the data. All the array values are stored in consecutive order.

If you were able to see an array on paper, it would look something like this:

[0] [1] [2]

In our example, the data looks like this:

["Heather"] ["Nikki"] ["Mich"]

Like other programming languages, the index is always a numerical value and the starting index is always 0. Just remember, index 0 is always the first element in an array. As you can see in the above example, we create the array by assigning the first index (0) a string value ("Heather").

The next two lines continue filling out the array, progressing through the index consecutively.

$userNames[1] = "Nikki";

The second array element (index 1) is assigned a different string value ("Nikki"), as is the third (index 2). At this point, we still have a
single array structure, but it is holding three separate values we can access. Excellent for organization.

The last section of code shows how you can access the data that has been stored in the array. Again, you use a numerical index to point to an element in the array. If you want to access the first element, use 0:

```bash
echo($userNames[0]);
```

In a later section, you will learn about looping structures that make using arrays a lot simpler. Before moving on, you should know that an array does not have to be a single, ordered list. TorqueScript also support multidimensional arrays.

An single-dimensional array contains a single row of values. A multidimensional array is essentially an array of arrays, which introduces columns as well. The following is a visual of what a multidimensional looks like with three rows and three columns:

```
[x] [x] [x]
[x] [x] [x]
[x] [x] [x]
```

Defining this kind of array in TorqueScript is simple. The following creates an array with 3 rows and 3 columns.

**Example:**

```bash
$testArray[0,0] = "a";
$testArray[0,1] = "b";
$testArray[0,2] = "c";
$testArray[1,0] = "d";
$testArray[1,1] = "e";
$testArray[1,2] = "f";
```
Notice that we are now using two indices, both starting at 0 and stopping at 2. We can use these as coordinates to determine which array element we are accessing:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

In our example, which progresses through the alphabet, you can visualize the data in the same way:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td>c</td>
</tr>
<tr>
<td>d</td>
<td>e</td>
<td>f</td>
</tr>
<tr>
<td>g</td>
<td>h</td>
<td>i</td>
</tr>
</tbody>
</table>

The first element [0,0] points to the letter 'a'. The last element [2,2] points to the letter 'i'.

```plaintext
$testArray[2,0] = "g";
$testArray[2,1] = "h";
$testArray[2,2] = "i";
```
**Vectors**

"Vectors" are a helpful data-type which are used throughout Torque 3D. For example, many fields in the World Editor take numeric values in sets of 3 or 4. These are stored as strings and interpreted as "vectors".

```
"1.0 1.0 1.0" (3 element vector)
```

The most common example of a vector would be a world position. Like most 3D coordinate systems, an object's position is stored as (X Y Z). You can use a three element vector to hold this data:

**Example:**

```
%position = "25.0 32 42.5";
```

You can separate the values using spaces or tabs (bot are acceptable whitespace). Another example is storing color data in a four element vector. The values that make up a color are "Red Blue Green Alpha," which are all numbers. You can create a vector for color using hard numbers, or variables:

**Example:**

```
%firstColor = "100 100 100 1.0";
echo(%firstColor);

%red = 128;
%blue = 255;
%green = 64;
%alpha = 1.0;

%secondColor = %red SPC %blue SPC %green SPC %alpha;
```

```echo(%secondColor);
```
Operations

Operators in TorqueScript behave very similarly to operators in real world math and other programming languages. You should recognize quite a few of these from math classes you took in school, but with small syntactical changes. The rest of this section will explain the syntax and show a brief example, but we will cover these in depth in later guides.
Arithmetic Operations

Basic math operations are supported using a syntax clear to everyone. TorqueScript supports multiplication, division, modulo, addition, and subtraction. Additionally, standard auto-increment and auto-decrement operations are available:

Example:

```
// Multiplication
%product = 3*4; // results in 12

// Division
%quotient = 4/2; // results in 2

// Modulo
%mod = 5%2; // results in 1

// Addition
%sum = 3+4; // results in 7

// Subtraction
%difference = 4-3; // results in 1

// Auto-increment
%value = 3;
%incr = %value++; // results in 4;

// Auto-decrement
%value = 3;
%decre = %value--; // results in 2;
```

See also:
Full Reference
Relational Operations

Relational operators are used for comparing values and variables against each other. Again, the syntax for these operations closely resemble real world math. The value returned from a comparison will always be true(1) or false(0).

Example:

```php
// Greater than
if(4 > 3)
    echo("True. 3 is not greater than 4");

// Greater than or equal to
if(3 >= 3)
    echo("True. 3 is equal to or greater than

// Equal to
if(6 == 6);
    echo("True. 6 is exactly equal to 6");

// Not equal to
if(3 != 5)
    echo("True. 3 is not equal to 5");
```

See also:
Full Reference
Bitwise Operations

Bitwise operations are used for comparing and shifting the bits of a value.

Examples:

```c
// Bitwise NOT/complement. Unary operation that flips bits.
%value = 101;
%bitValue = ~%value;  // results in 101 becoming 010

// Bitwise AND. When applied to two binary values, resulting bits are 1 if original pairs were 1.
%valueOne = 0101;
%valueTwo = 0011;
%bitValue = %valueOne&%valueTwo;  // Results in 0001
```
Assignment Operations

Assignment operators are used for setting the value of a variable. You should recognize it as the "equals" sign.

Example:

```plaintext
%val = 3; // Assigns the value of 3 to the %val variable
%val = 3+4; // Assigns the value of 3+4 to the %val variable
%val += 3; // Assigns the %val variable the value of 3 plus itself
```

See also:
Full Reference
String Operations

There are special values you can use to concatenate strings and variables. Concatenation refers to the joining of multiple values into a single variable. The following is the basic syntax:

"string 1" operation "string 2"

You can use string operators similarly to how you use mathematical operators (=, +, -, *). You have four operators at your disposal: @ NL TAB SPC. For example, the @ symbol will concatenate two strings together exactly how you specify, without adding any additional whitespace.

Example:

```plaintext
%newString = "Hello" @ "World";
echo(%newString);

// OUTPUT: HelloWorld
```

See also:
Full Reference
Control Statements

TorqueScript provides basic branching structures that will be familiar to programmers that have used other languages. If you are completely new to programming, you use branching structures to control your game's flow and logic. This section builds on everything you have learned about TorqueScript so far.

if, then, else

This type of structure is used to test a condition, then perform certain actions if the condition passes or fails. You do not always have to use the full structure, but the following syntax shows the extent of the conditional:

```
if(<boolean expression>)
{
    pass logic
}
else
{
    alternative logic
}
```

Remember how boolean values work? Essentially, a bool can either be true (1) or false (0). The condition (boolean) is always typed into the parenthesis after the "if" syntax. Your logic will be typed within the brackets {}. The following example uses specific variable names and conditions to show how this can be used:

Example:

```
// Global variable that controls lighting
$lightsShouldBeOn = true;
```
// Check to see if lights should be on or off
if($lightsShouldBeOn)
{
    // True. Call turn on lights function
    turnOnLights();

    echo("Lights have been turned on");
}
else
{
    // False. Turn off the lights
    turnOffLights();

    echo("Lights have been turned off");
}

Brackets for single line statements are optional. If you are thinking about adding additional logic to the code, then you should use the brackets anyway. If you know you will only use one logic statement, you can use the following syntax:

Example:

// Global variable that controls lighting
$lightsShouldBeOn = true;

// Check to see if lights should be on or off
if($lightsShouldBeOn)
    turnOnLights();  // True. Call turn on lig
else
    turnOffLights();  // False. Turn off the lig
**switch and switch$**

If your code is using several cascading if-then-else statements based on a single value, you might want to use a **switch** statement instead. Switch statements are easier to manage and read. There are two types of switch statements, based on data type: numeric (switch) and string (switch$).

**Switch Syntax:**

```plaintext
switch(<numeric expression>)
{
    case value0:
        statements;
    case value1:
        statements;
    case value3:
        statements;
    default:
        statements;
}
```

As the above code demonstrates, start by declaring the switch statement by passing in a value to the `switch(...)` line. Inside of the brackets {}, you will list out all the possible cases that will execute based on what value being tested. It is wise to always use the default case, anticipating rogue values being passed in.

**Example:**

```plaintext
switch($ammoCount)
{
    case 0:
        echo("Out of ammo, time to reload");
        reloadWeapon();
```
case 1:
echo("Almost out of ammo, warn user");
lowAmmoWarning();
case 100:
echo("Full ammo count");
playFullAmmoSound();
default:
doNothing();
}

switch only properly evaluates numerical values. If you need a switch statement to handle a string value, you will want to use switch$. The switch$ syntax is similar to what you just learned:

Switch$ Syntax:

```
switch$ (<string expression>)
{
    case "string value 0":
        statements;
    case "string value 1":
        statements;
    ...
    case "string value N":
        statements;
    default:
        statements;
}
```

Appending the $ sign to switch will immediately cause the parameter passed in to be parsed as a string. The following code applies this logic:

Example:
// Print out specialties
switch($userName)
{
    case "Heather":
        echo("Sniper");
    case "Nikki":
        echo("Demolition");
    case Mich:
        echo("Meat shield");
    default:
        echo("Unknown user");
}

Copyright © GarageGames, LLC. All Rights Reserved.
Loops

As the name implies, this structure type is used to repeat logic in a loop based on an expression. The expression is usually a set of variables that increase by count, or a constant variable changed once a loop has hit a specific point. For Loop
**For Loop**

for Loop Syntax:

```plaintext
for(expression0; expression1; expression2)
{
    statement(s);
}
```

One way to label the expressions in this syntax are (startExpression; testExpression; countExpression). Each expression is separated by a semi-colon.

**Example:**

```plaintext
for(%count = 0; %count < 3; %count++)
{
    echo(%count);
}
```

```plaintext
// OUTPUT:
// 0
// 1
// 2
```

The first expression creates the local variable count and initializing it to 0. In the second expression determines when to stop looping, which is when the count is no longer less than 3. Finally, the third expression increases the count the loop relies on.
**While Loop**

A while loop is a much simpler looping structure compared to a for loop.

**while Loop Syntax:**

```plaintext
while(expression)
{
    statements;
}
```

As soon as the expression is met, the while loop will terminate:

**Example:**

```plaintext
%countLimit = 0;
while(%countLimit <= 5)
{
    echo("Still in loop");
    %countLimit++;
}
echo("Loop was terminated");
```

// OUPUT:
// Still in loop
// Still in loop
// Still in loop
// Still in loop
// Still in loop
// Still in loop
// Still in loop
// Still in loop
// Loop was terminated
Functions

Much of your TorqueScript experience will come down to calling existing functions and writing your own. Functions are a blocks of code that only execute when you call them by name. Basic functions in TorqueScript are defined as follows:

```torquescript
// function - Is a keyword telling TorqueScript we are defining a new function.
// function_name - Is the name of the function we are creating.
// ... - Is any number of additional argument
// statements - Your custom logic executed when function is called
// return val - The value the function will give back after it has completed. Optional.

function function_name([arg0],...,[argn])
{
    statements;
    [return val;]
}
```

The `function` keyword, like other TorqueScript keywords, is case sensitive. You must type it exactly as shown above. The following is an example of a custom function that takes in two parameters, then executes code based on those arguments.

TorqueScript can take any number of arguments, as long as they are comma separated. If you call a function and pass fewer parameters than the function's definition specifies, the un-passed parameters will be given an empty string as their default value.

Example:

```torquescript
function echoRepeat (%echoString, %repeatCount)
{
    for (%count = 0; %count < %repeatCount; %c
    {
```
You can cause this function to execute by calling it in the console, or in another function:

```torquescript
echo(%echoString);
}
}
```

```torquescript
echoRepeat("hello!", 5);

// OUTPUT:
// "hello!"
// "hello!"
// "hello!"
// "hello!"
// "hello!"
```

If you define a function and give it the same name as a previously defined function, TorqueScript will completely override the old function. This still applies even if you change the number of parameters used; the older function will still be overridden.
Game Objects

The most complex aspect of TorqueScript involves dealing with game objects. Much of your object creation will be performed in the World Editor, but you should still know how to manipulate objects at a script level. One thing to remember is that everything in TorqueScript is an object: players, vehicles, items, etc.

Every object added in the level is saved to a mission file, which is written entirely in TorqueScript. This also means every game object is accessible from script. First, we will study the syntax of object creation.
Syntax

Even though objects are originally created in C++, they are exposed to script in a way that allows them to be declared using the following syntax:

**Object Definition:**

```torquescript
// In TorqueScript
%objectID = new ObjectType(Name : CopySource, {
  <datablock = DatablockIdentifier;>

  [existing_field0 = InitialValue0;]
  ...
  [existing_fieldN = InitialValueN;]

  [dynamic_field0 = InitialValue0;]
  ...
  [dynamic_fieldN = InitialValueN;]
};
```

**Syntax Breakdown:**

- **objectID** : Is the variable where the object's handle will be stored.

- **new** : Is a key word telling the engine to create an instance of the following ObjectType.

- **ObjectType** : Is any class declared in the engine or in script that has been derived from `SimObject` or a subclass of `SimObject`. SimObject-derived objects are what we were calling "game world objects" above.

- **Name (optional)** : Is any expression evaluating to a string,
which will be used as the object’s name.

- **CopySource (optional)**: The name of an object which is previously defined somewhere in script. Existing field values will be copied from CopySource to the new object being created. Any dynamic fields defined in CopySource will also be defined in the new object, and their values will be copied. Note: If CopySource is of a different ObjectType than the object being created, only CopySource’s dynamic fields will be copied.

- **arg0, ..., argn (optional)**: Is a comma separated list of arguments to the class constructor (if it takes any).

- **datablock**: Many objects (those derived from GameBase, or children of GameBase) require datablocks to initialize specific attributes of the new object. Datablocks are discussed below.

- **existing_fieldN**: In addition to initializing values with a datablock, you may also initialize existing class members (fields) here. Note: In order to modify a member of a C++-defined class, the member must be exposed to the Console. This concept is discussed in detail later.

- **dynamic_fieldN**: Lastly, you may create new fields (which will exist only in Script) for your new object. These will show up as dynamic fields in the World Editor Inspector.

The main object variants you can create are SimObjects without a datablock, and game objects which require a datablock. The most basic SimObject can be created in a single line of code:

**Example:**

```// Create a SimObject without any name, arguments
$exampleSimObject = new SimObject();```

The $exampleSimObject variable now has access to all the properties and functions of a basic SimObject. Usually, when you are creating a SimObject you will want custom fields to define
features

Example:

```java
// Create a SimObject with a custom field
$exampleSimObject = new SimObject()
{
    catchPhrase = "Hello world!";
};
```

As with the previous example, the above code creates a SimObject without a name which can be referenced by the global variable $exampleSimObject. This time, we have added a user defined field called "catchPhrase." There is not a single stock Torque 3D object that has a field called "catchPhrase." However, by adding this field to the SimObject it is now stored as long as that object exists.

The other game object variant mentioned previously involves the usage of datablocks. Datablocks contain static information used by a game object with a similar purpose. Datablocks are transmitted from a server to client, which means they cannot be modified while the game is running.

We will cover datablocks in more detail later, but the following syntax shows how to create a game object using a datablock.

Example:

```java
// create a StaticShape using a datablock
datablock StaticShapeData(ceiling_fan)
{
    category = "Misc";
    shapeFile = "art/shapes/undercity/cfan.dts"
    isInvincible = true;
};
```
new StaticShape(CistFan)
{
    dataBlock = "ceiling_fan";
    position = "12.5693 35.5857 59.5747";
    rotation = "1 0 0 0";
    scale = "1 1 1";
};

Once you have learned about datablocks, the process is quite simple:

1. Create a datablock in script, or using the datablock editor
2. Add a shape to the scene from script or using the World Editor
3. Assign the new object a datablock
Handles vs Names

Every game object added to a level can be accessed by two parameters:

- **Handle** - A unique numeric ID generated when the object is created
- **Name** - This is an optional parameter given to an object when it is created. You can assign a name to an object from the World Editor, or do so in TorqueScript using the following syntax:

Example:

```plaintext
// In this example, CistFan is the name of the object
new StaticShape(CistFan)
{
  dataBlock = "ceiling_fan";
  position = "12.5693 35.5857 59.5747";
  rotation = "1 0 0 0";
  scale = "1 1 1";
}
```

While in the World Editor, you will not be allowed to assign the same name to multiple, separate objects. The editor will ignore the attempt. If you manually name two objects the same thing in script, the game will only load the first object and ignore the second.
Singletons

If you need a global script object with only a single instance, you can use the singleton keyword. Singletons, in TorqueScript, are mostly used for unique shaders, materials, and other client-side only objects.

For example, SSAO (screen space ambient occlusion) is a post-processing effect. The game will only ever need a single instance of the shader, but it needs to be globally accessible on the client. The declaration of the SSAO shader in TorqueScript can be shown below:

```torque
singleton ShaderData( SSAOShader )
{
    DXVertexShaderFile = "shaders/common/postFx/postFxV.hlsl"
    DXPixelShaderFile = "shaders/common/postFx/ssao/SSAO_P.hlsl"
    pixVersion = 3.0;
};
```
Methods

In addition to the creation of stand-alone functions, TorqueScript allows you to create and call methods attached to objects. Some of the more important ConsoleMethods are already written in C++, then exposed to script. You can call these methods by using the dot (.) notation.

Syntax:

```
objHandle.function_name();
objName.function_name();
```

Example:

```
new StaticShape(CistFan)
{
    dataBlock = "ceiling_fan"
    position = "12.5693 35.5857 59.5747"
    rotation = "1 0 0 0"
    scale = "1 1 1"
};

// Write all the objects methods to the console
CistFan.dump();

// Get the ID of an object, using the object's name
$objID = CistFan.getID();

// Print the ID to the console
echo("Object ID: ", $objID);

// Get the object's position, using the object's handle
```
%position = $objID.getPosition();

// Print the position to the console
echo("Object Position: ", %position);

The above example shows how you can call an object's method by using its name or a variable containing its handle (unique ID number). Additionally, TorqueScript supports the creation of methods that have no associated C++ counterpart.

Syntax:

// function - Is a keyword telling TorqueScript we are defining a new function.
// ClassName::- Is the class type this function is supposed to work with.
// function_name - Is the name of the function we are creating.
// ... - Is any number of additional arguments.
// statements - Your custom logic executed when function is called
// %this- Is a variable that will contain the handle of the 'calling object'.
// return val - The value the function will give back after it has completed. Optional.
function ClassName::func_name(%this, [arg0],...,[argn])
{
    statements;
    [return val;]
}

At a minimum, object methods require that you pass them an object handle. You will often see the first argument named this. People use this as a hint, but you can name it anything you want. As with Console functions any number of additional arguments can be specified separated by commas.

As a simple example, let's say there is an object called Samurai, derived from the Player class. It is likely that a specific appearance and play style will be given to the samurai, so custom ConsoleMethods can be written. Here is a sample:
Example:

```plaintext
function Samurai::sheatheSword(%this)
{
    echo("Katana sheathed");
}
```

When you add a Samurai object to your level via the World Editor, it will be given an ID. Let's pretend the handle (ID number) is 1042. We can call its ConsoleMethod once it is defined, using the period syntax:

Example:

```plaintext
1042.sheatheSword();

// OUTPUT: "Katana sheathed"
```

Notice that no parameters were passed into the function. The this parameter is inherent, and the original function did not require any other parameters.
Packages

The package keyword tells the console that the subsequent block of code is to be declared but not loaded. Packages provide dynamic function-polymorphism in TorqueScript. In short, a function defined in a package will over-ride the prior definition of a same named function when the is activated. When the package is subsequently de-activated, the previous definition of any overridden functions will be re-asserted.

A package has the following syntax:

```
package package_name
{
  function function_definition0()
  {
    // code
  }

  function function_definitionN()
  {
    // code
  }
}
```

Some things to know:

- The same function can be defined in multiple packages.
- Only functions can be packaged.
- Datablocks cannot be packaged.
- Packages 'stack' meaning that deactivating packages activated prior to the currently active (s) will deactivate all packages activated prior to the being deactivated (see example below).
- Functions in a may activate and deactivate packages.

In order to use the functions in a package, the package must be
activated:

```
activatePackage(package_name);
```

Subsequently a package can be deactivated:

```
deactivatePackage(package_name);
```

**Usage**

First, define a function and two packages, each of which provides an alternative definition by the same name.

```
function testFunction()
{
    echo("testFunction() - unpackaged.");
}

package MyPackage0
{
    function testFunction()
    {
        echo("testFunction() - MyPackage0." );
    }
};

package MyPackage1
{
    function testFunction()
    {
        echo("testFunction() - MyPackage1." );
    }
};
```
Now invoke the testFunction() function from the console under three different conditions:

```cpp
==> testFunction();
testFunction() - unpackaged.

==> activatePackage( MyPackage0 );

==> testFunction();
testFunction() - MyPackage0.

==> activatePackage( MyPackage1 );

==> testFunction();
testFunction() - MyPackage1.

==> deactivatePackage( MyPackage0 ); // MyPa

==> testFunction();
testFunction() - unpackaged.
```

See also:

Packages
Quick Reference

- TorqueScript Quick Reference
  - Lexical Structure
  - Types
  - Statements
  - Expressions
  - TorqueScript Operators

Copyright © GarageGames, LLC. All Rights Reserved.
Lexical Structure

Comments

Single-line comments start with two forward slashes next to each other.

// Comment up to next newline.

Multi-line comments start with /* and end with */ (without the spaces between the symbols).

/* Multi-line comment */
Literals

Text
Keywords

The following identifiers are reserved in TorqueScript and cannot be used as names.

assert
break
case
continue
datablock
default
do
else
false
for
foreach
foreach$
function
if
in
namespace
new
NL
or
package
return
singleton
SPC
switch
switch$
TAB
true
while
Types

TorqueScript has four different kinds of types:

- Atomics
- Compounds
- Enumerations
- Bitfields
- Objects
Atomic Types

Atomic types are the building block types of TorqueScript. All literals have atomic types.

The following atomic types are defined:

**bool**

A boolean value. `true` or `false`.

**int**

A signed integer value between -2147483648 and 2147483647.

**float**

A single-precision floating-point value.

**string**

A string of UTF-8 encoded Unicode characters.
### Compound Types

A compound type is a type with structured values that are composed of individual values of atomic types.

The following compound types are defined:

**ColorI**

A RGBA color value in 8bit precision integer values. Each color component ranges from 0-255.

**ColorF**

A RGBA color value in single-precision floating-point values. Each color component ranges from 0-1.

**Point3F/VectorF**

**MatrixF**
Enumeration Types

Bitfield Types

Object Types

Copyright © GarageGames, LLC. All Rights Reserved.
Statements

```plaintext
statement :
    iteration_statement
  | branching_statement
  | block_statement
  | expression_statement
  | empty_statement
```
Iteration Statements

<table>
<thead>
<tr>
<th>iteration_statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>: for_statement</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

for Statement

while Statement

do Statement

foreach Statement

foreach_statement : 'foreach' '(' variable 'in' expression

The foreach statement iterates over all objects contained in a SimSet.

Example:

```
foreach( %obj in RootGroup )
  echo( %obj.getName() );
```

foreach$ Statement

foreachstr_statement : 'foreach$' '(' variable 'in' expression

The foreach$ statement iterates over all words in a string. Words are separated by newlines, spaces, and/or tabs.

Example:
Field Manipulators

```bash
foreach $( %str in "a b c d" )
    echo( %str );
```
Branching Statements

if Statement

```plaintext
if_statement
: 'if' '('< expression ')')' statement
| 'if' '('< expression ')')' statement 'else
```

switch Statement

switch Statement

return Statement
Expressions

Unary Expressions

Negation (prefix)  -
Increment (postfix)  ++
Decrement (postfix)  --
Binary Expressions

Numeric Inequality !=
Numeric Equality  ==
String Equality    $=
String Inequality  !$=
Lookup Expressions

Member Expression

Call Expression

Object Creation Expressions

Copyright © GarageGames, LLC. All Rights Reserved.
Operators in TorqueScript behave very similarly to operators in real world math and other programming languages. You should recognize quite a few of these from math classes you took in school, but with small syntactical changes. The rest of this section will explain the syntax and show a brief example, but we will cover these in depth in later guides.
# Arithmetic Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Name</th>
<th>Example</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>multiplication</td>
<td>$a \times b$</td>
<td>Multiply $a$ and $b$.</td>
</tr>
<tr>
<td>/</td>
<td>division</td>
<td>$a / b$</td>
<td>Divide $a$ by $b$.</td>
</tr>
<tr>
<td>%</td>
<td>modulo</td>
<td>$a % b$</td>
<td>Remainder of $a$ divided by $b$.</td>
</tr>
<tr>
<td>+</td>
<td>addition</td>
<td>$a + b$</td>
<td>Add $a$ and $b$.</td>
</tr>
<tr>
<td>-</td>
<td>subtraction</td>
<td>$a - b$</td>
<td>Subtract $b$ from $a$.</td>
</tr>
<tr>
<td>++</td>
<td>auto-increment</td>
<td>$a++$</td>
<td>Increment $a$. Note: $++a$ is illegal.</td>
</tr>
<tr>
<td></td>
<td>(post-fix only)</td>
<td></td>
<td>Note: the value of $a++$ is the incremented variable:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$a++$ is post-fix in syntax, but pre-increment in semantics.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Increment is post-fix in syntax, but pre-increment in semantics.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The variable is incremented, before return value is calculated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>This behavior is unlike that of C and C++.</td>
</tr>
<tr>
<td>- -</td>
<td>auto-decrement</td>
<td>$b--$</td>
<td>Decrement $b$. Note: $--b$ is illegal.</td>
</tr>
<tr>
<td></td>
<td>(post-fix only)</td>
<td></td>
<td>Note: the value of $a--$ is the decremented variable:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$b--$ is post-fix in syntax, but pre-decrement in semantics.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Decrement is post-fix in syntax, but pre-decrement in semantics.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The variable is decremented, before the return value is calculated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>This behavior is unlike that of C and C++.</td>
</tr>
<tr>
<td>Operator</td>
<td>Name</td>
<td>Example</td>
<td>Explanation</td>
</tr>
<tr>
<td>----------</td>
<td>-----------------------</td>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&lt;</td>
<td>Less than</td>
<td>$a &lt; b$</td>
<td>1 if $a$ is less than % $b$ (0 otherwise.)</td>
</tr>
<tr>
<td>&gt;</td>
<td>More than</td>
<td>$a &gt; b$</td>
<td>1 if $a$ is greater than % $b$ (0 otherwise.)</td>
</tr>
<tr>
<td>&lt;=</td>
<td>Less than or Equal to</td>
<td>$a &lt;= b$</td>
<td>1 if $a$ is less than or equal to % $b$ (0 otherwise.)</td>
</tr>
<tr>
<td>&gt;=</td>
<td>More than or Equal to</td>
<td>$a &gt;= b$</td>
<td>1 if $a$ is greater than or equal to % $b$ (0 otherwise.)</td>
</tr>
<tr>
<td>==</td>
<td>Equal to</td>
<td>$a == b$</td>
<td>1 if $a$ is equal to % $b$ (0 otherwise.)</td>
</tr>
<tr>
<td>!=</td>
<td>Not equal to</td>
<td>$a != b$</td>
<td>1 if $a$ is not equal to % $b$ (0 otherwise.)</td>
</tr>
<tr>
<td>!</td>
<td>Logical NOT</td>
<td>!$a$</td>
<td>1 if $a$ is 0 (0 otherwise.)</td>
</tr>
<tr>
<td>&amp;&amp;</td>
<td>Logical AND</td>
<td>$a &amp;&amp; b$</td>
<td>1 if $a$ and $b$ are both non-zero (0 otherwise.)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Logical OR</td>
</tr>
<tr>
<td>$=$</td>
<td>String equal to</td>
<td>$c == d$</td>
<td>1 if $c$ equal to $d$</td>
</tr>
<tr>
<td>!$=$</td>
<td>String not equal to</td>
<td>$c != d$</td>
<td>1 if $c$ not equal to $d$</td>
</tr>
</tbody>
</table>
## Bitwise Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Name</th>
<th>Example</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>~</td>
<td>Bitwise complement</td>
<td>~$a</td>
<td>flip bits 1 to 0 and 0 to 1. (i.e. ~10b == 01b)</td>
</tr>
<tr>
<td>&amp;</td>
<td>Bitwise AND</td>
<td>$a &amp; $b</td>
<td>composite of elements where bits in same position are 1. (i.e. 1b &amp; 1b == 1b)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>^</td>
<td>Bitwise XOR</td>
<td>$a ^ $b</td>
<td>composite of elements where bits in same position are opposite. (i.e. 100b &amp; 101b == 001b)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;&lt;</td>
<td>Left Shift</td>
<td>$a &lt;&lt; 3</td>
<td>element shifted left by 3 and padded with zeros. (i.e. 11b &lt;&lt; 3 == 11000b)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&gt;&gt;</td>
<td>Right Shift</td>
<td>$a &gt;&gt; 3</td>
<td>element shifted right by 3 and padded with zeros. (i.e. 110b &gt;&gt; 3d == 00011b)</td>
</tr>
</tbody>
</table>

- Bitwise complement: Flips bits 1 to 0 and 0 to 1.
- Bitwise AND: Composite of elements where bits in the same position are 1.
- Bitwise OR: Composite of elements where bits 1 in either of the two elements.
- Bitwise XOR: Composite of elements where bits in the same position are opposite.
- Left Shift: Element shifted left by 3 and padded with zeros.
- Right Shift: Element shifted right by 3 and padded with zeros.
### Assignment Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Name</th>
<th>Example</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
<td>Assignment</td>
<td>$a = b;</td>
<td>Assign value of $b to $a. Note: the value of an assignment is the value being assigned, so $a = b = c$ is legal.</td>
</tr>
<tr>
<td>op=</td>
<td>Assignment Operators</td>
<td>$a op= b;</td>
<td>Equivalent to $a = a op b$ for $op$ can be any of: $/ % + - &amp;</td>
</tr>
</tbody>
</table>
## String Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Name</th>
<th>Example</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>@</td>
<td>String concatenation</td>
<td>$c @ d$</td>
<td>Concatenates strings $c$ and $d$ into a single string. Numeric literals/variables convert to strings.</td>
</tr>
</tbody>
</table>
| NL       | New Line         | $c \text{NL} d$ | Concatenates strings $c$ and $d$ into a single string separated by new-line.  
**Note:** such a string can be decomposed with getRecord(). |
| TAB      | Tab              | $c \text{TAB} d$ | Concatenates strings $c$ and $d$ into a single string separated by tab.  
**Note:** such a string can be decomposed with getField(). |
| SPC      | Space            | $c \text{SPC} d$ | Concatenates strings $c$ and $d$ into a single string separated by space.  
**Note:** such a string can be decomposed with getWord(). |
Modules

Here is a list of all modules:

- **Camera System**
  - Base Camera
  - Path Camera
- **Core**
  - Console
    - Debugging
    - Logging
    - Messaging
    - Packages
    - Scripting
  - File I/O
    - File Searching
  - Math
    - Matrix Math
    - Random Numbers
    - Vector Math
  - Strings
    - Field Manipulators
  - Utilities
- **Environment Objects**
  - Atmosphere
  - Foliage
  - Forest
  - Miscellaneous
  - Terrain
  - Water
- **Examples**
- **GUI**
  - 3D Controls
  - Button Controls
  - Container Controls
  - Core Controls
  - Game Controls
  - General Controls
- **Image and Video Controls**
- **Utility Controls**
- **Value Controls**

**Game**
- **AI**
- **Game Objects**
- **Miscellaneous**
- **Physics**
- **Special Effects**
  - **Decals**
- **Vehicles**

**Input Management**
- **Input Event Listing**

**Networking**

**Platform**
- **Localization**
- **Mac**
- **Windows**

**Rendering**
- **Font**
- **GFX**
  - **Materials**
  - **Shaders**
- **Lighting**
  - **Advanced Lighting**
  - **Basic Lighting**
- **Render Binning**

**Section**

**Sound**
- **FMOD**

**UNDOCUMENTED**

Copyright © GarageGames, LLC. All Rights Reserved.
Camera System

This section is dedicated to the various camera objects in Torque 3D. More...
<table>
<thead>
<tr>
<th>Modules</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Base Camera</strong></td>
</tr>
<tr>
<td>The base camera object that is typically manipulated by a GameConnection's input.</td>
</tr>
<tr>
<td><strong>Path Camera</strong></td>
</tr>
<tr>
<td>A camera that moves along a path.</td>
</tr>
</tbody>
</table>
## Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void setDefaultFov(float defaultFOV)</code></td>
<td>Set the default FOV for a camera.</td>
</tr>
<tr>
<td><code>void setFov(float FOV)</code></td>
<td>Set the FOV of the camera.</td>
</tr>
<tr>
<td><code>void setZoomSpeed(int speed)</code></td>
<td>Set the zoom speed of the camera. This affects how quickly the camera changes from one field of view to another.</td>
</tr>
</tbody>
</table>
Detailed Description

This section is dedicated to the various camera objects in Torque 3D.
Function Documentation

void setDefaultFov(float defaultFOV )

Set the default FOV for a camera.

**Parameters:**

*defaultFOV* The default field of view in degrees

void setFov(float FOV )

Set the FOV of the camera.

**Parameters:**

*FOV* The camera's new FOV in degrees

void setZoomSpeed(int speed )

Set the zoom speed of the camera. This affects how quickly the camera changes from one field of view to another.

**Parameters:**

*speed* The camera's zoom speed in ms per 90deg FOV change

Copyright © GarageGames, LLC. All Rights Reserved.
Base Camera
[Camera System]

The base camera object that is typically manipulated by a GameConnection's input. More...
## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Camera</strong></td>
<td>Represents a position, direction and field of view to render a scene from. [More...]</td>
</tr>
<tr>
<td><strong>CameraData</strong></td>
<td>A datablock that describes a camera. [More...]</td>
</tr>
</tbody>
</table>
Enumerations

```c
enum CameraMotionMode { Stationary, FreeRotate, Fly, OrbitObject, OrbitPoint, TrackObject, Overhead, EditOrbit }
```

Movement behavior type for Camera.

More...
### Variables

<table>
<thead>
<tr>
<th>static float</th>
<th>Camera::movementSpeed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Global camera movement speed in units/s (typically m/s), with a base value of 40.</td>
</tr>
</tbody>
</table>
**Detailed Description**

The base camera object that is typically manipulated by a GameConnection's input.
Enumeration Type Documentation

def CameraMotionMode

Movement behavior type for Camera.

**Enumerator:**

- **Stationary**  
  Camera does not rotate or move.

- **FreeRotate**  
  Camera may rotate but does not move.

- **Fly**  
  Camera may rotate and move freely.

- **OrbitObject**  
  Camera orbits about a given object. Damage flash and white out is determined by the object being orbited. See Camera::setOrbitMode() to set the orbit object and other parameters.

- **OrbitPoint**  
  Camera orbits about a given point. See Camera::setOrbitMode() to set the orbit point and other parameters.

- **TrackObject**  
  Camera always faces a given object. See Camera::setTrackObject() to set the object to track and a distance to remain from the object.

- **Overhead**  
  Camera moves in the XY plane.

- **EditOrbit**  
  Used by the World Editor to orbit about a point. When first activated, the camera is rotated to face the orbit point rather than move to it.
Variable Documentation

**float Camera::movementSpeed** [static, inherited]

Global camera movement speed in units/s (typically m/s), with a base value of 40.

Used in the following camera modes:

- Edit Orbit Mode
- Fly Mode
- Overhead Mode
Path Camera
[Camera System]

A camera that moves along a path. More...
<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PathCamera</strong></td>
<td>A camera that moves along a path. The camera can then be made to travel along this path forwards or backwards.</td>
</tr>
<tr>
<td><strong>PathCameraData</strong></td>
<td>General interface to control a PathCamera object from the script level.</td>
</tr>
</tbody>
</table>
Detailed Description

A camera that moves along a path.
Core

Basic engine and language functionality for TorqueScript. More...
# Modules

<table>
<thead>
<tr>
<th>Utilities</th>
<th>Miscellaneous utility functions.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Console</td>
<td>The basis of the TorqueScript system and command execution.</td>
</tr>
<tr>
<td>File I/O</td>
<td>Functions allowing you to search for files, read them, write them, and access their properties.</td>
</tr>
<tr>
<td>Strings</td>
<td>Functions for dealing with string values.</td>
</tr>
<tr>
<td>Math</td>
<td>Functions for dealing with vectors and matrices etc.</td>
</tr>
</tbody>
</table>
Detailed Description

Basic engine and language functionality for TorqueScript.
Console
[Core]

The basis of the TorqueScript system and command execution.

More...
## Classes

<table>
<thead>
<tr>
<th>class</th>
<th><strong>ScriptGroup</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Essentially a <strong>SimGroup</strong>, but with onAdd and onRemove script callbacks. <a href="#">More...</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th><strong>ScriptObject</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A script-level OOP object which allows binding of a class, superClass and arguments along with declaration of methods. <a href="#">More...</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th><strong>SimGroup</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A collection of SimObjects that are owned by the group. <a href="#">More...</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th><strong>SimObject</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Base class for almost all objects involved in the simulation. <a href="#">More...</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th><strong>SimSet</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A collection of SimObjects. <a href="#">More...</a></td>
</tr>
</tbody>
</table>
## Modules

<table>
<thead>
<tr>
<th><strong>Logging</strong></th>
<th>Functions for logging messages, warnings, and errors to the console.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Debugging</strong></td>
<td>Functionality to help spot program errors.</td>
</tr>
<tr>
<td><strong>Scripting</strong></td>
<td>Functions for working with script code.</td>
</tr>
<tr>
<td><strong>Packages</strong></td>
<td>Functions relating to the control of packages.</td>
</tr>
<tr>
<td><strong>Messaging</strong></td>
<td>Script classes and functions used for passing messages and events between classes.</td>
</tr>
</tbody>
</table>
## Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td><code>cls ()</code></td>
<td>Clears the console output.</td>
</tr>
<tr>
<td>void</td>
<td><code>debugEnumInstances (string className, string functionName)</code></td>
<td>Call the given function for each instance of the given class.</td>
</tr>
<tr>
<td>bool</td>
<td><code>dumpEngineDocs (string outputFile)</code></td>
<td>Dumps the engine scripting documentation to the specified file overwriting any existing content.</td>
</tr>
<tr>
<td>SimXMLDocument</td>
<td><code>exportEngineAPIToXML ()</code></td>
<td>Create a XML document containing a dump of the entire exported engine API.</td>
</tr>
<tr>
<td>string</td>
<td><code>getCategoryOfClass (string className)</code></td>
<td>Returns the category of the given class.</td>
</tr>
<tr>
<td>string</td>
<td><code>getDescriptionOfClass (string className)</code></td>
<td>Returns the description string for the named class.</td>
</tr>
<tr>
<td>bool</td>
<td><code>isClass (string identifier)</code></td>
<td>Returns true if the passed identifier is the name of a declared class.</td>
</tr>
<tr>
<td>bool</td>
<td><code>isMemberOfClass (string className, string superClassName)</code></td>
<td>Returns true if the class is derived from the super class.</td>
</tr>
<tr>
<td>bool</td>
<td><code>isValidObjectName (string name)</code></td>
<td>Return true if the given name makes for a valid object name.</td>
</tr>
<tr>
<td>SimObject</td>
<td><code>loadObject (string filename)</code></td>
<td>Loads a serialized object from a file.</td>
</tr>
</tbody>
</table>
### bool saveObject (SimObject object, string filename)
 Serialize the object to a file.

### void unitTest_runTests ([$ searchString[, bool skipInteractive]]))
 Run unit tests, or just the tests that prefix match against the searchString.
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>$instantGroup&lt;br&gt;The group that objects will be added to when they are created.</td>
</tr>
<tr>
<td>bool</td>
<td>$Con::alwaysUseDebugOutput&lt;br&gt;Determines whether to send output to the platform's &quot;debug&quot; system.</td>
</tr>
<tr>
<td>bool</td>
<td>$Con::logBufferEnabled&lt;br&gt;    If true, the log buffer will be enabled.</td>
</tr>
<tr>
<td>int</td>
<td>$Con::objectCopyFailures&lt;br&gt;   If greater than zero then it counts the number of object creation failures based on a missing copy object and does not report an error.</td>
</tr>
<tr>
<td>int</td>
<td>$Con::printLevel&lt;br&gt;          This is deprecated.</td>
</tr>
<tr>
<td>bool</td>
<td>$Con::useTimestamp&lt;br&gt;        If true a timestamp is prepended to every console message.</td>
</tr>
<tr>
<td>bool</td>
<td>$Con::warnUndefinedVariables&lt;br&gt;If true, a warning will be displayed in the console whenever a undefined variable is used in script.</td>
</tr>
</tbody>
</table>
Detailed Description

The basis of the TorqueScript system and command execution.
Function Documentation

void cls( )

Clears the console output.

void debugEnumInstances(string className, string functionName)

Call the given function for each instance of the given class.

Parameters:

  className  Name of the class for which to enumerate instances.

  functionName  Name of function to call and pass each instance of the given class.

Note:

This function is only available in debug builds and primarily meant as an aid in debugging.

bool dumpEngineDocs(string outputFile)

Dumps the engine scripting documentation to the specified file overwriting any existing content.

Parameters:

  outputFile  The relative or absolute output file path and name.

Returns:

  Returns true if successful.
SimXMLDocument exportEngineAPIToXML( )

Create a XML document containing a dump of the entire exported engine API.

**Returns:**

A SimXMLDocument containing a dump of the engine's export information or NULL if the operation failed.

string getCategoryOfClass (string className )

Returns the category of the given class.

**Parameters:**

*className*  The name of the class.

string getDescriptionOfClass (string className )

Returns the description string for the named class.

**Parameters:**

*className*  The name of the class.

**Returns:**

The class description in string format.

bool isClass (string identifier )

Returns true if the passed identifier is the name of a declared class.

bool isMemberOfClass (string className,
string superClassName
)

Returns true if the class is derived from the super class.
If either class doesn't exist this returns false.

**Parameters:**

- `className` The class name.
- `superClassName` The super class to look for.

bool isValidObjectName (string name)

Return true if the given name makes for a valid object name.

**Parameters:**

- `name` Name of object

**Returns:**

True if name is allowed, false if denied (usually because it starts with a number, _, or invalid character

SimObject loadObject (string filename)

Loads a serialized object from a file.

**Parameters:**

- `Name` and path to text file containing the object

bool saveObject (SimObject object, string filename)


Serialize the object to a file.

**Parameters:**

- *object*  The object to serialize.
- *filename*  The file name and path.

```plaintext
void unitTest_runTests([searchString[, bool skipInteractive]] )
```

Run unit tests, or just the tests that prefix match against the searchString.
Variable Documentation

string $instantGroup

The group that objects will be added to when they are created.

bool $Con::alwaysUseDebugOutput

Determines whether to send output to the platform's "debug" system.

Note:
This is disabled in shipping builds.

bool $Con::logBufferEnabled

If true, the log buffer will be enabled.

int $Con::objectCopyFailures

If greater than zero then it counts the number of object creation failures based on a missing copy object and does not report an error..

int $Con::printLevel

This is deprecated.
It is no longer in use and does nothing.

bool $Con::useTimestamp
If true a timestamp is prepended to every console message.

```cpp
bool $Con::warnUndefinedVariables
```

If true, a warning will be displayed in the console whenever a undefined variable is used in script.
Debugging
[Console]

Functionality to help spot program errors. More...
### Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void backtrace ()</code></td>
<td>Prints the scripting call stack to the console log.</td>
</tr>
<tr>
<td><code>void debug ()</code></td>
<td>Drops the engine into the native C++ debugger.</td>
</tr>
<tr>
<td><code>void debugDumpAllObjects ()</code></td>
<td>Dumps all current EngineObject instances to the console.</td>
</tr>
<tr>
<td><code>void debugv (string variableName)</code></td>
<td>Logs the value of the given variable to the console.</td>
</tr>
<tr>
<td><code>void dumpAlloc (int allocNum)</code></td>
<td>Dumps information about the given allocated memory block.</td>
</tr>
<tr>
<td><code>void dumpMemSnapshot (string fileName)</code></td>
<td>Dumps a snapshot of current memory to a file.</td>
</tr>
<tr>
<td><code>void dumpUnflaggedAllocs (string fileName=&quot;&quot;)</code></td>
<td>Dumps all unflagged memory allocations.</td>
</tr>
<tr>
<td><code>void flagCurrentAllocs ()</code></td>
<td>Flags all current memory allocations.</td>
</tr>
<tr>
<td><code>void freeMemoryDump ()</code></td>
<td>Dumps some useful statistics regarding free memory.</td>
</tr>
<tr>
<td><code>void profilerDump ()</code></td>
<td>Dumps current profiling stats to the console window.</td>
</tr>
<tr>
<td><code>void profilerDumpToFile (string fileName)</code></td>
<td>Dumps current profiling stats to a file.</td>
</tr>
<tr>
<td><code>void profilerEnable (bool enable)</code></td>
<td>Enables or disables the profiler.</td>
</tr>
<tr>
<td><code>void profilerMarkerEnable (string markerName, bool enable=true)</code></td>
<td>Enable or disable a specific profile.</td>
</tr>
<tr>
<td><code>void profilerReset ()</code></td>
<td>Resets the profiler, clearing it of all its data.</td>
</tr>
<tr>
<td>Function</td>
<td>Parameters</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>int sizeof (string objectOrClass)</code></td>
<td></td>
</tr>
<tr>
<td><code>void telnetSetParameters (int port, string consolePass, string listenPass, bool remoteEcho=false)</code></td>
<td></td>
</tr>
<tr>
<td><code>void trace (bool enable=true)</code></td>
<td></td>
</tr>
<tr>
<td><code>void validateMemory ()</code></td>
<td></td>
</tr>
</tbody>
</table>
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td><code>getBuildString</code></td>
<td>Get the type of build, &quot;Debug&quot; or &quot;Release&quot;.</td>
</tr>
<tr>
<td>string</td>
<td><code>getCompileTimeString</code></td>
<td>Get the time of compilation.</td>
</tr>
<tr>
<td>string</td>
<td><code>getEngineName</code></td>
<td>Get the name of the engine product that this is running from, as a string.</td>
</tr>
<tr>
<td>int</td>
<td><code>getVersionNumber</code></td>
<td>Get the version of the build, as a string.</td>
</tr>
<tr>
<td>string</td>
<td><code>getVersionString</code></td>
<td>Get the version of the build, as a string.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Functionality to help spot program errors.

Also provides profiler functions, helpful in determining performance bottlenecks.
Function Documentation

void backtrace( )

Prints the scripting call stack to the console log.

Used to trace functions called from within functions. Can help discover what functions were called (and not yet exited) before the current point in scripts.

void debug( )

Drops the engine into the native C++ debugger.

This function triggers a debug break and drops the process into the IDE’s debugger. If the process is not running with a debugger attached it will generate a runtime error on most platforms.

Note:
This function is not available in shipping builds.

void debugDumpAllObjects( )

Dumps all current EngineObject instances to the console.

Note:
This function is only available in debug builds.

void debugv(string variableName )

Logs the value of the given variable to the console.

Prints a string of the form "<variableName> = <variable value>" to
the console.

**Parameters:**

 variableName Name of the local or global variable to print.

**Example:**

```c
%var = 1;
debugv("%var"); // Prints "%var = 1"
```

```c
void dumpAlloc(int allocNum)
```

Dumps information about the given allocated memory block.

**Parameters:**

 allocNum Memory block to dump information about.

**Note:**

Available in debug builds only. In torqueConfig.h, TORQUE_DISABLE_MEMORY_MANAGER must be undefined to use this function.

```c
void dumpMemSnapshot(string fileName)
```

Dumps a snapshot of current memory to a file.

The total memory used will also be output to the console. This function will attempt to create the file if it does not already exist.

**Parameters:**

 fileName Name and path of file to save profiling stats to.
 Must use forward slashes (/)

**Example:**
dumpMemSnapshot("C:/Torque/ProfilerLogs/\t\t")

Note:
Available in debug builds only. In torqueConfig.h, TORQUE_DISABLE_MEMORY_MANAGER must be undefined to use this function.

void dumpUnflaggedAllocs(string fileName = "")

Dumps all unflagged memory allocations.

Dumps all memory allocations that were made after a call to flagCurrentAllocs(). Helpful when used with flagCurrentAllocs() for detecting memory leaks and analyzing general memory usage.

Parameters:
fileName Optional file path and location to dump all memory allocations not flagged by flagCurrentAllocs(). If left blank, data will be dumped to the console.

Example:
dumpMemSnapshot(); // dumps info to console
dumpMemSnapshot("C:/Torque/profilerlog1.txt")

Note:
Available in debug builds only. In torqueConfig.h, TORQUE_DISABLE_MEMORY_MANAGER must be undefined to use this function.

void flagCurrentAllocs()

Flags all current memory allocations.

Flags all current memory allocations for exclusion in subsequent
calls to `dumpUnflaggedAllocs()`. Helpful in detecting memory leaks and analyzing memory usage.

```c
void freeMemoryDump( )
```

Dumps some useful statistics regarding free memory.

Dumps an analysis of 'free chunks' of memory. Does not print how much memory is free.

```c
void profilerDump( )
```

Dumps current profiling stats to the console window.

**Note:**

Markers disabled with `profilerMarkerEnable()` will be skipped over. If the profiler is currently running, it will be disabled.

```c
void profilerDumpToFile( string fileName )
```

Dumps current profiling stats to a file.

**Note:**

If the profiler is currently running, it will be disabled.

**Parameters:**

(fileName) Name and path of file to save profiling stats to.

(fileName) Must use forward slashes (/). Will attempt to create the file if it does not already exist.

**Example:**

```c
profilerDumpToFile( "C:/Torque/log1.txt" );
```
void profilerEnable(bool enable)

Enables or disables the profiler.

Data is only gathered while the profiler is enabled.

**Note:**
Profiler is not available in shipping builds. T3D has predefined profiling areas surrounded by markers, but you may need to define additional markers (in C++) around areas you wish to profile, by using the PROFILE_START( markerName ); and PROFILE_END(); macros.

void profilerMarkerEnable(string markerName, bool enable = true)

Enable or disable a specific profile.

**Parameters:**
- `enable` Optional parameter to enable or disable the profile.
- `markerName` Name of a specific marker to enable or disable.

**Note:**
Calling this function will first call profilerReset(), clearing all data from profiler. All profile markers are enabled by default.

void profilerReset()

Resets the profiler, clearing it of all its data.

If the profiler is currently running, it will first be disabled. All markers will retain their current enabled/disabled status.
```c
int sizeof(string objectOrClass )
```

Determines the memory consumption of a class or object.

**Parameters:**

- `objectOrClass` The object or class being measured.

**Returns:**

Returns the total size of an object in bytes.

```c
void telnetSetParameters(int port,  
                          string consolePass,  
                          string listenPass,  
                          bool remoteEcho = false )
```

Initializes and open the telnet console.

**Parameters:**

- `port` Port to listen on for console connections (0 will shut down listening).
- `consolePass` Password for read/write access to console.
- `listenPass` Password for read access to console.
- `remoteEcho` [optional] Enable echoing back to the client, off by default.

```c
void trace(bool enable = true )
```

Enable or disable tracing in the script code VM.

When enabled, the script code runtime will trace the invocation and returns from all functions that are called and log them to the console. This is helpful in observing the flow of the script program.
Parameters:

*enable* New setting for script trace execution, on by default.

```c
void validateMemory()
```

Used to validate memory space for the game.
## Variable Documentation

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>string getBuildString</code></td>
<td>Get the type of build, &quot;Debug&quot; or &quot;Release&quot;.</td>
</tr>
<tr>
<td><code>string getCompileTimeString</code></td>
<td>Get the time of compilation.</td>
</tr>
<tr>
<td><code>string getEngineName</code></td>
<td>Get the name of the engine product that this is running from, as a string.</td>
</tr>
<tr>
<td><code>int getVersionNumber</code></td>
<td>Get the version of the build, as a string.</td>
</tr>
<tr>
<td><code>string getVersionString</code></td>
<td>Get the version of the build, as a string.</td>
</tr>
</tbody>
</table>
Logging
[Console]

Functions for logging messages, warnings, and errors to the console. More...
Classes

```java
class ConsoleLogger
```
Enumerations

```c
enum LogLevel {
    normal,
    warning,
    error
}
```

Priority levels for logging entries.

More...
## Functions

<table>
<thead>
<tr>
<th>void</th>
<th><strong>dumpConsoleClasses</strong> (bool dumpScript=true, bool dumpEngine=true)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Dumps all declared console classes to the console.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th><strong>dumpConsoleFunctions</strong> (bool dumpScript=true, bool dumpEngine=true)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Dumps all declared console functions to the console.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th><strong>echo</strong> (string message...)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Logs a message to the console.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th><strong>error</strong> (string message...)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Logs an error message to the console.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th><strong>log</strong> (string message)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Logs a message to the console.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th><strong>logError</strong> (string message)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Logs an error message to the console.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th><strong>logWarning</strong> (string message)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Logs a warning message to the console.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th><strong>setLogMode</strong> (int mode)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Determines how log files are written.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th><strong>warn</strong> (string message...)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Logs a warning message to the console.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Functions for logging messages, warnings, and errors to the console.
Enumeration Type Documentation

enum LogLevel

Priority levels for logging entries.

**Enumerator:**

*normal*  Lowest priority level, no highlighting.

*warning* Mid level priority, tags and highlights possible issues in blue.

*error* Highest priority level, extreme emphasis on this entry. Highlighted in red.
Function Documentation

void dumpConsoleClasses(bool dumpScript = true, bool dumpEngine = true)

Dumps all declared console classes to the console.

Parameters:

dumpScript   Optional parameter specifying whether or not classes defined in script should be dumped.

dumpEngine   Optional parameter specifying whether or not classes defined in the engine should be dumped.

void dumpConsoleFunctions(bool dumpScript = true, bool dumpEngine = true)

Dumps all declared console functions to the console.

Parameters:

dumpScript   Optional parameter specifying whether or not functions defined in script should be dumped.

dumpEngine   Optional parameter specifying whether or not functions defined in the engine should be dumped.

void echo(string message...)

Logs a message to the console.

Concatenates all given arguments to a single string and prints the string to the console. A newline is added automatically after the
void error (string message... )

Logs an error message to the console.

Concatenates all given arguments to a single string and prints the string to the console as an error message (in the in-game console, these will show up using a red font by default). A newline is added automatically after the text.

Parameters:

message Any number of string arguments.

void log (string message )

Logs a message to the console.

Parameters:

message The message text.

Note:

By default, messages will appear white in the console.

void logError (string message )

Logs an error message to the console.

Parameters:

message The message text.
Note:
By default, errors will appear red in the console.

```java
void logWarning (string message)
```

Logs a warning message to the console.

**Parameters:**
- `message` The message text.

Note:
By default, warnings will appear turquoise in the console.

```java
void setLogMode (int mode)
```

Determines how log files are written.

Sets the operational mode of the console logging system.

**Parameters:**
- `mode` Parameter specifying the logging mode. This can be:
  - 1: Open and close the console log file for each separate string of output. This will ensure that all parts get written out to disk and that no parts remain in intermediate buffers even if the process crashes.
  - 2: Keep the log file open and write to it continuously. This will make the system operate faster but if the process crashes, parts of the output may not have been written to disk yet and will be missing from the log.

Additionally, when changing the log mode and thus opening a new log file, either of the two mode values may be combined by binary OR with 0x4 to cause the logging system to flush all console log
messages that had already been issued to the console system into the newly created log file.

**Note:**

Xbox 360 does not support logging to a file. Use Platform::OutputDebugStr in C++ instead.

```cpp
void warn(string message... )
```

Logs a warning message to the console.

Concatenates all given arguments to a single string and prints the string to the console as a warning message (in the in-game console, these will show up using a turquoise font by default). A newline is added automatically after the text.

**Parameters:**

- `message` Any number of string arguments.
Messaging
(Console)

Script classes and functions used for passing messages and events between classes. More...
### Classes

<table>
<thead>
<tr>
<th>class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>EventManager</strong></td>
<td>The <strong>EventManager</strong> class is a wrapper for the standard messaging system. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>Message</strong></td>
<td>Base class for messages. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>MessageForwarder</strong></td>
<td>Forward messages from one queue to another. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>ScriptMsgListener</strong></td>
<td>Script accessible version of Dispatcher::IMessageListener. Often used in conjunction with <strong>EventManager</strong>. <a href="#">More...</a></td>
</tr>
</tbody>
</table>
## Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function Name</th>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>dispatchMessage</code></td>
<td>(string queueName, string message, string data)</td>
<td>Dispatch a message to a queue.</td>
</tr>
<tr>
<td>bool</td>
<td><code>dispatchMessageObject</code></td>
<td>(string queueName, string message)</td>
<td>Dispatch a message object to a queue.</td>
</tr>
<tr>
<td>bool</td>
<td><code>isQueueRegistered</code></td>
<td>(string queueName)</td>
<td>Determines if a dispatcher queue exists.</td>
</tr>
<tr>
<td>bool</td>
<td><code>registerMessageListener</code></td>
<td>(string queueName, string listener)</td>
<td>Registers an event message.</td>
</tr>
<tr>
<td>void</td>
<td><code>registerMessageQueue</code></td>
<td>(string queueName)</td>
<td>Registers a dispatcher queue.</td>
</tr>
<tr>
<td>void</td>
<td><code>unregisterMessageListener</code></td>
<td>(string queueName, string listener)</td>
<td>Unregisters an event message.</td>
</tr>
<tr>
<td>void</td>
<td><code>unregisterMessageQueue</code></td>
<td>(string queueName)</td>
<td>Unregisters a dispatcher queue.</td>
</tr>
</tbody>
</table>
Detailed Description

Script classes and functions used for passing messages and events between classes.
Function Documentation

```csharp
bool dispatchMessage(string queueName, string message, string data)
```

Dispatch a message to a queue.

**Parameters:**
- `queueName` Queue to dispatch the message to
- `message` Message to dispatch
- `data` Data for message

**Returns:**
True for success, false for failure

**See also:**
- `dispatchMessageObject`

```csharp
bool dispatchMessageObject(string queueName, string message)
```

Dispatch a message object to a queue.

**Parameters:**
- `queueName` Queue to dispatch the message to
- `message` Message to dispatch

**Returns:**
true for success, false for failure

**See also:**
dispatchMessage

bool isQueueRegistered (string queueName )

Determines if a dispatcher queue exists.

**Parameters:**

- `queueName` String containing the name of queue

bool registerMessageListener (string queueName, string listener )

Registers an event message.

**Parameters:**

- `queueName` String containing the name of queue to attach listener to
- `listener` Name of event messenger

void registerMessageQueue (string queueName )

Registers a dispatcher queue.

**Parameters:**

- `queueName` String containing the name of queue

void unregisterMessageListener (string queueName, string listener )

Unregisters an event message.
Parameters:

queueName  String containing the name of queue
listener   Name of event messenger

void unregisterMessageQueue ( string queueName )

Unregisters a dispatcher queue.

Parameters:

queueName  String containing the name of queue
Packages
[Console]

Functions relating to the control of packages. More...
## Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td><code>activatePackage (String packageName)</code></td>
<td>Activates an existing package.</td>
</tr>
<tr>
<td>void</td>
<td><code>deactivatePackage (String packageName)</code></td>
<td>Deactivates a previously activated package.</td>
</tr>
<tr>
<td>string</td>
<td><code>getFunctionPackage (string funcName)</code></td>
<td>Provides the name of the package the function belongs to.</td>
</tr>
<tr>
<td>string</td>
<td><code>getMethodPackage (string namespace, string method)</code></td>
<td>Provides the name of the package the method belongs to.</td>
</tr>
<tr>
<td>string</td>
<td><code>getPackageList ()</code></td>
<td>Returns a space delimited list of the active packages in stack order.</td>
</tr>
<tr>
<td>bool</td>
<td><code>isPackage (String identifier)</code></td>
<td>Returns true if the identifier is the name of a declared package.</td>
</tr>
</tbody>
</table>
Detailed Description

Functions relating to the control of packages.

See also:

Packages
Function Documentation

void activatePackage (String packageName )

Activates an existing package.

The activation occurs by updating the namespace linkage of existing functions and methods. If the package is already activated the function does nothing.

void deactivatePackage (String packageName )

Deactivates a previously activated package.

The package is deactivated by removing its namespace linkages to any function or method. If there are any packages above this one in the stack they are deactivated as well. If the package is not on the stack this function does nothing.

string getFunctionPackage (string funcName )

Provides the name of the package the function belongs to.

Parameters:

   funcName  String containing name of the function

Returns:

   The name of the function's package

string getMethodPackage (string namespace, string method )
Provides the name of the package the method belongs to.

**Parameters:**
- *namespace* Class or namespace, such as `Player`
- *method* Name of the function to search for

**Returns:**
The name of the method's package

```csharp
string getPackageList()
```

Returns a space delimited list of the active packages in stack order.

```csharp
bool isPackage(String identifier)
```

Returns true if the identifier is the name of a declared package.
Scripting

[Console]

Functions for working with script code. More...
## Classes

<table>
<thead>
<tr>
<th>class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ArrayObject</strong></td>
<td>Data structure for storing indexed sequences of key/value pairs. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>ScriptGroup</strong></td>
<td>Essentially a <strong>SimGroup</strong>, but with onAdd and onRemove script callbacks. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>ScriptObject</strong></td>
<td>A script-level OOP object which allows binding of a class, superClass and arguments along with declaration of methods. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>SimGroup</strong></td>
<td>A collection of SimObjects that are owned by the group. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>SimSet</strong></td>
<td>A collection of SimObjects. <a href="#">More...</a></td>
</tr>
</tbody>
</table>
## Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>string call (string functionName, string args...)</code></td>
<td>Apply the given arguments to the specified global function and return the result of the call.</td>
</tr>
<tr>
<td><code>bool compile (string fileName, bool overrideNoDSO=false)</code></td>
<td>Compile a file to bytecode.</td>
</tr>
<tr>
<td><code>void deleteVariables (string pattern)</code></td>
<td>Undefine all global variables matching the given name <em>pattern</em>.</td>
</tr>
<tr>
<td><code>bool exec (string fileName, bool noCalls=false, bool journalScript=false)</code></td>
<td>Execute the given script file.</td>
</tr>
<tr>
<td><code>bool execPrefs (string relativeFileName, bool noCalls=false, bool journalScript=false)</code></td>
<td>Manually execute a special script file that contains game or editor preferences.</td>
</tr>
<tr>
<td><code>void export (string pattern, string filename=&quot;&quot;, bool append=false)</code></td>
<td>Write out the definitions of all global variables matching the given name <em>pattern</em>.</td>
</tr>
<tr>
<td><code>string getDSOPath (string scriptFileName)</code></td>
<td>Get the absolute path to the file in which the compiled code for the given script file will be stored.</td>
</tr>
<tr>
<td><code>string getVariable (string varName)</code></td>
<td>Returns the value of the named variable or an empty string if not found.</td>
</tr>
<tr>
<td><code>bool isDefined (string varName)</code></td>
<td>Determines if a variable exists and contains a value.</td>
</tr>
<tr>
<td><code>bool isFunction (string funcName)</code></td>
<td>Determines if a function exists or not.</td>
</tr>
<tr>
<td><code>bool isMethod (string namespace, string method)</code></td>
<td>Determines if a class/namespace method exists.</td>
</tr>
<tr>
<td>void   setVariable (string varName, string value)</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Sets the value of the named variable.</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

Functions for working with script code.
**Function Documentation**

```plaintext
string call (string functionName, string args...)
```

Apply the given arguments to the specified global function and return the result of the call.

**Parameters:**

- `functionName` The name of the function to call. This function must be in the global namespace, i.e. you cannot call a function in a namespace through `call`. Use `eval()` for that.

**Returns:**

The result of the function call.

**Example:**

```plaintext
function myFunction( %arg )
{
    return ( %arg SPC "World!" );
}

echo( call( "myFunction", "Hello" ) ); //
```

```plaintext
bool compile (string fileName, bool overrideNoDSO = false)
```

Compile a file to bytecode.

This function will read the TorqueScript code in the specified file, compile it to internal bytecode, and, if DSO generation is enabled
or `overrideNoDDSO` is true, will store the compiled code in a `.dso` file in the current DSO path mirroring the path of `fileName`.

**Parameters:**

- `fileName`  
  *Path* to the file to compile to bytecode.  
  If true, force generation of DSOs even if the `overrideNoDDSO` engine is compiled to not generate write compiled code to DSO files.

**Returns:**

True if the file was successfully compiled, false if not.

**Note:**

The definitions contained in the given file will not be made available and no code will actually be executed. Use `exec()` for that.

**See also:**

- `getDSOPath`
- `exec`

```java
void deleteVariables(string pattern )
```

Undefine all global variables matching the given name `pattern`.

**Parameters:**

- `pattern`  
  A global variable name pattern. Must begin with '$'.

**Example:**

```java
// Define a global variable in the "My" name
$My::Variable = "value";

// Undefine all variable in the "My" names
deleteVariables( "$My::*" );
```
bool exec(string fileName,
    bool noCalls = false,
    bool journalScript = false
)

Execute the given script file.

**Parameters:**
- *fileName* Path to the file to execute
- *noCalls* Deprecated
- *journalScript* Deprecated

**Returns:**
True if the script was successfully executed, false if not.

**Example:**
```c
define
    // Execute the init.cs script file found in the same directory as the current script file.
    exec( "./init.cs" );
```

**See also:**
- compile
- eval

bool execPrefs(string relativeFileName,
    bool noCalls = false,
    bool journalScript = false
)

Manually execute a special script file that contains game or editor preferences.

**See also:**
- strIsMatchExpr
Parameters:

- **relativeFileName** Name and path to file from project folder
- **noCalls** Deprecated
- **journalScript** Deprecated

Returns:

True if script was successfully executed

Note:

Appears to be useless in Torque 3D, should be deprecated

```cpp
void export(string pattern,
             string filename = "",
             bool append = false
)
```

Write out the definitions of all global variables matching the given name `pattern`.

If `fileName` is not "", the variable definitions are written to the specified file. Otherwise the definitions will be printed to the console.

The output are valid TorqueScript statements that can be executed to restore the global variable values.

Parameters:

- **pattern** A global variable name pattern. Must begin with '$'.
- **filename** Path of the file to which to write the definitions or "" to write the definitions to the console.
  
  If true and `fileName` is not "", then the definitions are appended to the specified file. Otherwise existing contents of the file (if any) will be overwritten.

Example:
string getDSOPath(string scriptFileName)

Get the absolute path to the file in which the compiled code for the given script file will be stored.

**Parameters:**

- `scriptFileName` Path to the .cs script file.

**Returns:**

The absolute path to the .dso file for the given script file.

**Note:**

The compiler will store newly compiled DSOs in the prefs path but pre-existing DSOs will be loaded from the current paths.

**See also:**

- `compile`
- `getPrefsPath`

string getVariable(string varName)

Returns the value of the named variable or an empty string if not found.

**Name of the variable to search for**

**Returns:**

Value contained by varName, "" if the variable does not exist

bool isDefined(string varName)

Determines if a variable exists and contains a value.

**Parameters:**

`varName` Name of the variable to search for

**Returns:**

True if the variable was defined in script, false if not

**Example:**

```isDefined( "$myVar" );```

---

`bool isFunction(string funcName )`

Determines if a function exists or not.

**Parameters:**

`funcName` String containing name of the function

**Returns:**

True if the function exists, false if not

---

`bool isMethod(string namespace, string method )`

Determines if a class/namespace method exists.

**Parameters:**

`namespace` Class or namespace, such as Player
`method` Name of the function to search for

**Returns:**

True if the method exists, false if not
void setVariable(string varName, string value)

Sets the value of the named variable.

**Parameters:**
- *varName* Name of the variable to locate
- *value* New value of the variable

**Returns:**
- True if variable was successfully found and set
File I/O
[Core]

Functions allowing you to search for files, read them, write them, and access their properties. More...
## Classes

<table>
<thead>
<tr>
<th>class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FileDialog</strong></td>
<td>Base class responsible for displaying an OS file browser. More...</td>
</tr>
<tr>
<td><strong>FileObject</strong></td>
<td>This class is responsible opening, reading, creating, and saving file contents. More...</td>
</tr>
<tr>
<td><strong>FileStreamObject</strong></td>
<td>A wrapper around StreamObject for parsing text and data from files. More...</td>
</tr>
<tr>
<td><strong>OpenFileDialog</strong></td>
<td>Derived from FileDialog, this class is responsible for opening a file browser with the intention of opening a file. More...</td>
</tr>
<tr>
<td><strong>OpenFolderDialog</strong></td>
<td>OS level dialog used for browsing folder structures. More...</td>
</tr>
<tr>
<td><strong>SaveFileDialog</strong></td>
<td>Derived from FileDialog, this class is responsible for opening a file browser with the intention of saving a file. More...</td>
</tr>
<tr>
<td><strong>SimXMLDocument</strong></td>
<td>File I/O object used for creating, reading, and writing XML documents. More...</td>
</tr>
<tr>
<td><strong>StreamObject</strong></td>
<td>Base class for working with streams. More...</td>
</tr>
<tr>
<td><strong>ZipObject</strong></td>
<td>Provides access to a zip file. More...</td>
</tr>
</tbody>
</table>
Modules

File Searching

Functions for searching files by name patterns.
### Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>createPath</code> (string path)</td>
<td>Create the given directory or the path leading to the given filename.</td>
</tr>
<tr>
<td>string</td>
<td><code>expandFilename</code> (string filename)</td>
<td>Grabs the full path of a specified file.</td>
</tr>
<tr>
<td>string</td>
<td><code>expandOldFilename</code> (string filename)</td>
<td>Retrofits a filepath that uses old Torque style.</td>
</tr>
<tr>
<td>String</td>
<td><code>fileBase</code> (string fileName)</td>
<td>Get the base of a file name (removes extension).</td>
</tr>
<tr>
<td>String</td>
<td><code>fileCreatedTime</code> (string fileName)</td>
<td>Returns a platform specific formatted string with the creation time for the file.</td>
</tr>
<tr>
<td>bool</td>
<td><code>fileDelete</code> (string path)</td>
<td>Delete a file from the hard drive.</td>
</tr>
<tr>
<td>String</td>
<td><code>fileExt</code> (string fileName)</td>
<td>Get the extension of a file.</td>
</tr>
<tr>
<td>String</td>
<td><code>fileModifiedTime</code> (string fileName)</td>
<td>Returns a platform specific formatted string with the last modified time for the file.</td>
</tr>
<tr>
<td>String</td>
<td><code>fileName</code> (string fileName)</td>
<td>Get the file name of a file (removes extension and path).</td>
</tr>
<tr>
<td>String</td>
<td><code>filePath</code> (string fileName)</td>
<td>Get the path of a file (removes name and extension).</td>
</tr>
<tr>
<td>int</td>
<td><code>fileSize</code> (string fileName)</td>
<td>Determines the size of a file on disk.</td>
</tr>
<tr>
<td>String</td>
<td><code>getGetCurrentDirectory</code> ()</td>
<td>Return the current working directory.</td>
</tr>
<tr>
<td>String</td>
<td><code>getDirectoryList</code> (string path, int depth=0)</td>
<td>Gathers a list of directories starting at the given path.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td><code>getExecutableName()</code></td>
<td>Gets the name of the game's executable.</td>
<td></td>
</tr>
<tr>
<td><code>getIntFileCRC(string fileName)</code></td>
<td>Provides the CRC checksum of the given file.</td>
<td></td>
</tr>
<tr>
<td><code>getMainDotCsDir()</code></td>
<td>Get the absolute path to the directory that contains the main.cs script from which the engine was started.</td>
<td></td>
</tr>
<tr>
<td><code>getWorkingDirectory()</code></td>
<td>Reports the current directory.</td>
<td></td>
</tr>
<tr>
<td><code>IsDirectory(string directory)</code></td>
<td>Determines if a specified directory exists or not.</td>
<td></td>
</tr>
<tr>
<td><code>isFile(string fileName)</code></td>
<td>Determines if the specified file exists or not.</td>
<td></td>
</tr>
<tr>
<td><code>isWriteableFileName(string fileName)</code></td>
<td>Determines if a file name can be written to using File I/O.</td>
<td></td>
</tr>
<tr>
<td><code>makeFullPath(string path, string cwd='')</code></td>
<td>Converts a relative file path to a full path.</td>
<td></td>
</tr>
<tr>
<td><code>makeRelativePath(string path, string to='')</code></td>
<td>Turns a full or local path to a relative one.</td>
<td></td>
</tr>
<tr>
<td><code>openFile(string file)</code></td>
<td>Open the given file through the system. This will usually open the file in its associated application.</td>
<td></td>
</tr>
<tr>
<td><code>openFolder(string path)</code></td>
<td>Open the given folder in the system's file manager.</td>
<td></td>
</tr>
<tr>
<td><code>pathConcat(string path, string file)</code></td>
<td>Combines two separate strings containing a file path and file name together into a single string.</td>
<td></td>
</tr>
<tr>
<td><code>pathCopy(string fromFile, string toFile, bool noOverwrite=true)</code></td>
<td>Copy a file to a new location.</td>
<td></td>
</tr>
<tr>
<td><code>setCurrentDirectory(string path)</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Set the current working directory.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>void startFileChangeNotifications ()</code></td>
<td>Start watching resources for file changes.</td>
<td></td>
</tr>
<tr>
<td><code>void stopFileChangeNotifications ()</code></td>
<td>Stop watching resources for file changes.</td>
<td></td>
</tr>
</tbody>
</table>
## Variables

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>string</strong></td>
<td>$Con::File</td>
</tr>
<tr>
<td></td>
<td>The currently executing script file.</td>
</tr>
<tr>
<td><strong>string</strong></td>
<td>$Con::Root</td>
</tr>
<tr>
<td></td>
<td>The mod folder for the currently executing script file.</td>
</tr>
</tbody>
</table>
Detailed Description

Functions allowing you to search for files, read them, write them, and access their properties.
Function Documentation

bool createPath(string path)

Create the given directory or the path leading to the given filename.

If path ends in a trailing slash, then all components in the given path will be created as directories (if not already in place). If path, does not end in a trailing slash, then the last component of the path is taken to be a file name and only the directory components of the path will be created.

**Parameters:**

*path* The path to create.

**Note:**

Only present in a Tools build of Torque.

string expandFilename(string filename)

Grabs the full path of a specified file.

**Parameters:**

*filename* Name of the local file to locate

**Returns:**

String containing the full filepath on disk

string expandOldFilename(string filename)

Retrofits a filepath that uses old Torque style.

**Returns:**
String containing filepath with new formatting

**String fileBase (string fileName )**

Get the base of a file name (removes extension).

**Parameters:**

- `fileName` Name and path of file to check

**Returns:**

String containing the file name, minus extension

**String fileCreatedTime (string fileName )**

Returns a platform specific formatted string with the creation time for the file.

**Parameters:**

- `fileName` Name and path of file to check

**Returns:**

Formatted string (OS specific) containing created time, "9/3/2010 12:33:47 PM" for example

**bool fileDelete (string path )**

Delete a file from the hard drive.

**Parameters:**

- `path` Name and path of the file to delete

**Note:**

THERE IS NO RECOVERY FROM THIS. Deleted file is gone
for good.

**Returns:**
True if file was successfully deleted

---

**String fileExt (string fileName )**

Get the extension of a file.

**Parameters:**
*fileName* Name and path of file

**Returns:**
String containing the extension, such as ".exe" or ".cs"

---

**String fileModifiedTime (string fileName )**

Returns a platform specific formatted string with the last modified time for the file.

**Parameters:**
*fileName* Name and path of file to check

**Returns:**
Formatted string (OS specific) containing modified time, "9/3/2010 12:33:47 PM" for example

---

**String fileName (string fileName )**

Get the file name of a file (removes extension and path).

**Parameters:**
*fileName* Name and path of file to check
Returns:
String containing the file name, minus extension and path

String filePath (string fileName )

Get the path of a file (removes name and extension).

Parameters:
fileName Name and path of file to check

Returns:
String containing the path, minus name and extension

int fileSize (string fileName )

Determines the size of a file on disk.

Parameters:
fileName Name and path of the file to check

Returns:
Returns filesize in KB, or -1 if no file

String getCurrentDirectory ( )

Return the current working directory.

Returns:
The absolute path of the current working directory.

Note:
Only present in a Tools build of Torque.
See also:
getWorkingDirectory()

String getDirectoryList(String path, int depth = 0)

Gathers a list of directories starting at the given path.

Parameters:
  - **path**: String containing the path of the directory
  - **depth**: Depth of search, as in how many subdirectories to parse through

Returns:
  Tab delimited string containing list of directories found during search, """" if no files were found

String getExecutableName()

Gets the name of the game's executable.

Returns:
  String containing this game's executable name

int getFileCRC(String fileName)

Provides the CRC checksum of the given file.

Parameters:
  - **fileName**: The path to the file.

Returns:
The calculated CRC checksum of the file, or -1 if the file could not be found.

String getMainDotCsDir()

Get the absolute path to the directory that contains the main.cs script from which the engine was started.

This directory will usually contain all the game assets and, in a user-side game installation, will usually be read-only.

Returns:
The path to the main game assets.

String getWorkingDirectory()

Reports the current directory.

Returns:
String containing full file path of working directory

bool IsDirectory(string directory)

Determines if a specified directory exists or not.

Parameters:
directory String containing path in the form of "foo/bar"

Returns:
Returns true if the directory was found.

Note:
Do not include a trailing slash '/'.

bool isFile(string fileName)

Determines if the specified file exists or not.

**Parameters:**

- `fileName` The path to the file.

**Returns:**

Returns true if the file was found.

bool isWriteableFileName(string fileName)

Determines if a file name can be written to using File I/O.

**Parameters:**

- `fileName` Name and path of file to check

**Returns:**

Returns true if the file can be written to.

String makeFullPath(string path, string cwd = "")

Converts a relative file path to a full path.

For example, "./console.log" becomes "C:/Torque/t3d/examples/FPS Example/game/console.log"

**Parameters:**

- `path` Name of file or path to check
- `cwd` Optional current working directory from which to build the full path.
Returns:
String containing non-relative directory of path

String makeRelativePath(string path,
    string to = ""
  )

Turns a full or local path to a relative one.
For example, "./game/art" becomes "game/art"

Parameters:
  path Full path (may include a file) to convert
to Optional base path used for the conversion. If not supplied the current working directory is used.

Returns:
String containing relative path

void openFile(string file )

Open the given file through the system. This will usually open the file in its associated application.

Parameters:
  file Path of the file to open.

Note:
Only present in a Tools build of Torque.

void openFolder(string path )

Open the given folder in the system's file manager.
Parameters:

\textit{path} full path to a directory.

\textbf{Note:}

Only present in a Tools build of Torque.

\begin{verbatim}
String pathConcat(string path, 
    string file
)
\end{verbatim}

Combines two separate strings containing a file path and file name together into a single string.

Parameters:

- \textit{path} String containing file path
- \textit{file} String containing file name

Returns:

String containing concatenated file name and path

\begin{verbatim}
bool pathCopy(string fromFile, 
    string toFile, 
    bool noOverwrite = true
)
\end{verbatim}

Copy a file to a new location.

Parameters:

- \textit{fromFile} Path of the file to copy.
- \textit{toFile} Path where to copy \textit{fromFile} to.
- \textit{noOverwrite} If true, then \textit{fromFile} will not overwrite a file that may already exist at \textit{toFile}.

Returns:
True if the file was successfully copied, false otherwise.

**Note:**
Only present in a Tools build of Torque.

```c
bool setCurrentDirectory(string path)
```

Set the current working directory.

**Parameters:**
The absolute or relative (to the current working `path` directory) path of the directory which should be made the new working directory.

**Returns:**
True if the working directory was successfully changed to `path`, false otherwise.

**Note:**
Only present in a Tools build of Torque.

```c
void startFileChangeNotifications()
```

Start watching resources for file changes.
Typically this is called during `initializeCore()`.

**See also:**
`stopFileChangeNotifications()`

```c
void stopFileChangeNotifications()
```

Stop watching resources for file changes.
Typically this is called during shutdownCore().

See also:

startFileChangeNotifications()
Variable Documentation

string $Con::File

The currently executing script file.

string $Con::Root

The mod folder for the currently executing script file.
File Searching
[File I/O]
Functions for searching files by name patterns. More...
## Functions

<table>
<thead>
<tr>
<th>String</th>
<th><code>findFirstFile</code> (string pattern, bool recurse=true)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Returns the first file in the directory system matching the given pattern.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>String</th>
<th><code>findFirstFileMultiExpr</code> (string pattern, bool recurse=true)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Returns the first file in the directory system matching the given patterns.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>String</th>
<th><code>findNextFile</code> (string pattern=&quot;&quot;)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Returns the next file matching a search begun in <code>findFirstFile()</code>.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>String</th>
<th><code>findNextFileMultiExpr</code> (string pattern=&quot;&quot;)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Returns the next file matching a search begun in <code>findFirstFileMultiExpr()</code>.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int</th>
<th><code>getFileCount</code> (string pattern, bool recurse=true)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Returns the number of files in the directory tree that match the given patterns.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int</th>
<th><code>getFileCountMultiExpr</code> (string pattern, bool recurse=true)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Returns the number of files in the directory tree that match the given patterns.</td>
</tr>
</tbody>
</table>
Detailed Description

Functions for searching files by name patterns.
Function Documentation

```csharp
String findFirstFile(string pattern,
                     bool   recurse = true
)
```

Returns the first file in the directory system matching the given pattern.

Use the corresponding `findFirstFile()` to step through the results. If you're only interested in the number of files returned by the pattern match, use `getFileCount()`.

This function differs from `findFirstFileMultiExpr()` in that it supports a single search pattern being passed in.

**Note:**

You cannot run multiple simultaneous file system searches with these functions. Each call to `findFirstFile()` and `findFirstFileMultiExpr()` initiates a new search and renders a previous search invalid.

**Parameters:**

- `pattern` The path and file name pattern to match against. If true, the search will exhaustively recurse into subdirectories of the given path and match the given filename pattern.

**Returns:**

The path of the first file matched by the search or an empty string if no matching file could be found.

**Example:**

```csharp
// Execute all .cs files in a subdirectory
for( %file = findFirstFile( "subdirectory", exec( %file ) );
```
See also:

findNextFile()
getFileCount()
findFirstFileMultiExpr()

String findFirstFileMultiExpr(string pattern,
    bool recurse = true)

Returns the first file in the directory system matching the given patterns.

Use the corresponding findNextFileMultiExpr() to step through the results. If you're only interested in the number of files returned by the pattern match, use getFileCountMultiExpr().

This function differs from findFirstFile() in that it supports multiple search patterns to be passed in.

Note:

You cannot run multiple simultaneous file system searches with these functions. Each call to findFirstFile() and findFirstFileMultiExpr() initiates a new search and renders a previous search invalid.

Parameters:

- **pattern**
  - The path and file name pattern to match against, such as *.cs. Separate multiple patterns with TABs. For example: "*.cs" TAB "*.dso"
  - If true, the search will exhaustively recurse into subdirectories of the given path and match the given filename patterns.

Returns:

- String of the first matching file path, or an empty string if no matching files were found.
// Find all DTS or Collada models
%filePatterns = "*.dts" TAB "*.dae";
%fullPath = findFirstFileMultiExpr( %filePatterns );
while ( %fullPath !$= "" )
{
    echo( %fullPath );
    %fullPath = findNextFileMultiExpr( %filePatterns );
}

See also:
findNextFileMultiExpr()
getFileSizeMultiExpr()
findFirstFile()

String findNextFile(string pattern = "" )

Returns the next file matching a search begun in findFirstFile().

Parameters:

    The path and file name pattern to match against. This
    pattern is optional and may be left out as it is not used by the
    code. It is here for legacy reasons.

Returns:

    The path of the next filename matched by the search or an
    empty string if no more files match.

Example:

// Execute all .cs files in a subdirectory
for( %file = findFirstFile( "subdirectory", exec( %file ) );
See also:

findFirstFile()

String findNextFileMultiExpr(string pattern = "")

Returns the next file matching a search begun in findFirstFileMultiExpr().

Parameters:

The path and file name pattern to match against. This pattern is optional and may be left out as it is not used by the code. It is here for legacy reasons.

Returns:

String of the next matching file path, or an empty string if no matching files were found.

Example:

```c++
// Find all DTS or Collada models
%filePatterns = "*.dts" TAB "*.dae";
%fullPath = findFirstFileMultiExpr( %filePatterns
while ( %fullPath !$= "" )
{
    echo( %fullPath );
    %fullPath = findNextFileMultiExpr( %filePatterns
}
```

See also:

findFirstFileMultiExpr()

int getFileCount(string pattern, bool recurse = true)
Returns the number of files in the directory tree that match the given patterns.

This function differs from `getFileCountMultiExpr()` in that it supports a single search pattern being passed in.

If you're interested in a list of files that match the given pattern and not just the number of files, use `findFirstFile()` and `findNextFile()`.

**Parameters:**

- `pattern` The path and file name pattern to match against. If true, the search will exhaustively recurse into subdirectories of the given path and match the given filename pattern counting files in subdirectories.

**Returns:**

- Number of files located using the pattern

**Example:**

```csharp
// Count the number of .cs files in a subdirectory
getFileCount( "subdirectory/*.cs" );
```

**See also:**

- `findFirstFile()`
- `findNextFile()`
- `getFileCountMultiExpr()`

```csharp
int getFileCountMultiExpr (string pattern,
    bool   recurse = true
) {

    Returns the number of files in the directory tree that match the given patterns.

    If you're interested in a list of files that match the given patterns and not just the number of files, use `findFirstFileMultiExpr()` and
findNextFileMultiExpr().

Parameters:
The path and file name pattern to match against, such as *.cs. Separate multiple patterns with TABs. For example: "*.cs" TAB "*.dso"
If true, the search will exhaustively recurse into subdirectories of the given path and match the given filename pattern.

Returns:
Number of files located using the patterns

Example:

```plaintext
// Count all DTS or Collada models
%filePatterns = "*.dts" TAB "*.dae";
echo( "Number of shape files:" SPC getFileCountMultiExpr %filePatterns);
```

See also:
findFirstFileMultiExpr()
findNextFileMultiExpr()

Copyright © GarageGames, LLC. All Rights Reserved.
Math
[Core]

Functions for dealing with vectors and matrices etc. More...
## Modules

### Vector Math

Functions for working with three-dimensional vectors (VectorF/Point3F).

### Matrix Math

Functions for working with matrices (MatrixF, AngAxisF, MatrixRotation, MatrixPosition).

### Random Numbers

Functions for generating random numbers.
## Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point3F</td>
<td><code>getBoxCenter</code> (Box3F box)</td>
<td>Get the center point of an axis-aligned box.</td>
</tr>
<tr>
<td>float</td>
<td><code>getMax</code> (float v1, float v2)</td>
<td>Calculate the greater of two specified numbers.</td>
</tr>
<tr>
<td>float</td>
<td><code>getMin</code> (float v1, float v2)</td>
<td>Calculate the lesser of two specified numbers.</td>
</tr>
<tr>
<td>float</td>
<td><code>m2Pi</code> ()</td>
<td>Return the value of 2*PI (full-circle in radians).</td>
</tr>
<tr>
<td>float</td>
<td><code>mAbs</code> (float v)</td>
<td>Calculate absolute value of specified value.</td>
</tr>
<tr>
<td>float</td>
<td><code>mAcos</code> (float v)</td>
<td>Calculate the arc-cosine of v.</td>
</tr>
<tr>
<td>float</td>
<td><code>mAsin</code> (float v)</td>
<td>Calculate the arc-sine of v.</td>
</tr>
<tr>
<td>float</td>
<td><code>mAtan</code> (float rise, float run)</td>
<td>Calculate the arc-tangent (slope) of a line defined by rise and run.</td>
</tr>
<tr>
<td>void</td>
<td><code>mathInit</code> (...)</td>
<td>Install the math library with specified extensions.</td>
</tr>
<tr>
<td>int</td>
<td><code>mCeil</code> (float v)</td>
<td>Round v up to the nearest integer.</td>
</tr>
<tr>
<td>float</td>
<td><code>mClamp</code> (float v, float min, float max)</td>
<td>Clamp the specified value between two bounds.</td>
</tr>
<tr>
<td>float</td>
<td><code>mCos</code> (float v)</td>
<td>Calculate the cosine of v.</td>
</tr>
<tr>
<td>float</td>
<td><code>mDegToRad</code> (float degrees)</td>
<td>Convert specified degrees into radians.</td>
</tr>
<tr>
<td>string</td>
<td><code>mFloatLength</code> (float v, int precision)</td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><code>int mFloor (float v)</code></td>
<td>Round v down to the nearest integer.</td>
<td></td>
</tr>
<tr>
<td><code>float mFMod (float v, float d)</code></td>
<td>Calculate the remainder of v/d.</td>
<td></td>
</tr>
<tr>
<td><code>bool mIsPow2 (int v)</code></td>
<td>Returns whether the value is an exact power of two.</td>
<td></td>
</tr>
<tr>
<td><code>float mLerp (float v1, float v2, float time)</code></td>
<td>Calculate linearly interpolated value between two specified numbers using specified normalized time.</td>
<td></td>
</tr>
<tr>
<td><code>float mLog (float v)</code></td>
<td>Calculate the natural logarithm of v.</td>
<td></td>
</tr>
<tr>
<td><code>float mPi ()</code></td>
<td>Return the value of PI (half-circle in radians).</td>
<td></td>
</tr>
<tr>
<td><code>float mPow (float v, float p)</code></td>
<td>Calculate b raised to the p-th power.</td>
<td></td>
</tr>
<tr>
<td><code>float mRadToDeg (float radians)</code></td>
<td>Convert specified radians into degrees.</td>
<td></td>
</tr>
<tr>
<td><code>int mRound (float v)</code></td>
<td>Round v to the nearest integer.</td>
<td></td>
</tr>
<tr>
<td><code>float mSaturate (float v)</code></td>
<td>Clamp the specified value between 0 and 1 (inclusive).</td>
<td></td>
</tr>
<tr>
<td><code>float mSin (float v)</code></td>
<td>Calculate the sine of v.</td>
<td></td>
</tr>
<tr>
<td><code>string mSolveCubic (float a, float b, float c, float d)</code></td>
<td>Solve a cubic equation (3rd degree polynomial) of form a<em>x^3 + b</em>x^2 + c*x + d = 0.</td>
<td></td>
</tr>
<tr>
<td><code>string mSolveQuadratic (float a, float b, float c)</code></td>
<td>Solve a quadratic equation (2nd degree polynomial) of form a<em>x^2 + b</em>x + c = 0.</td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>mSolveQuartic</strong> (float a, float b, float c, float d, float e)</td>
<td>Solve a quartic equation (4th degree polynomial) of form $a\cdot x^4 + b\cdot x^3 + c\cdot x^2 + d\cdot x + e = 0$.</td>
<td></td>
</tr>
<tr>
<td>float <strong>mSqrt</strong> (float v)</td>
<td>Calculate the square-root of v.</td>
<td></td>
</tr>
<tr>
<td>float <strong>mTan</strong> (float v)</td>
<td>Calculate the tangent of v.</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description
Functions for dealing with vectors and matrices etc.
Function Documentation

Point3F getBoxCenter(Box3F box)

Get the center point of an axis-aligned box.

Parameters:

- A Box3F, in string format using "minExtentX minExtentY minExtentZ maxExtentX maxExtentY maxExtentZ"

Returns:

Center of the box.

float getMax(float v1, float v2)

Calculate the greater of two specified numbers.

Parameters:

- v1 Input value.
- v2 Input value.

Returns:

The greater value of the two specified values.

float getMin(float v1, float v2)

Calculate the lesser of two specified numbers.

Parameters:
v1  Input value.
v2  Input value.

**Returns:**
The lesser value of the two specified values.

```c
float m2Pi()
```

Return the value of 2*PI (full-circle in radians).

**Returns:**
The value of 2*PI.

```c
float mAbs(float v)
```

Calculate absolute value of specified value.

**Parameters:**

- v  Input Value.

**Returns:**
Absolute value of specified value.

```c
float mAcos(float v)
```

Calculate the arc-cosine of v.

**Parameters:**

- v  Input Value (in radians).

**Returns:**
The arc-cosine of the input value.
**float mAsin (float v )**

Calculate the arc-sine of v.

**Parameters:**
- v Input Value (in radians).

**Returns:**
- The arc-sine of the input value.

**float mAtan (float rise, float run )**

Calculate the arc-tangent (slope) of a line defined by rise and run.

**Parameters:**
- rise of line.
- run of line.

**Returns:**
- The arc-tangent (slope) of a line defined by rise and run.

**void mathInit ( ... )**

Install the math library with specified extensions.

Possible parameters are:

- 'DETECT' Autodetect math lib settings.
- 'C' Enable the C math routines. C routines are always enabled.
• 'FPU' Enable floating point unit routines.
• 'MMX' Enable MMX math routines.
• '3DNOW' Enable 3dNow! math routines.
• 'SSE' Enable SSE math routines.

```c
int mCeil(float v )
```

Round v up to the nearest integer.

**Parameters:**

- \(v\) Number to convert to integer.

**Returns:**

Number converted to integer.

```c
float mClamp(float v, float min, float max )
```

Clamp the specified value between two bounds.

**Parameters:**

- \(v\) Input value.
- \(min\) Minimum Bound.
- \(max\) Maximum Bound.

**Returns:**

The specified value clamped to the specified bounds.

```c
float mCos(float v )
```
Calculate the cosine of \( v \).

**Parameters:**
- \( v \) Input Value (in radians).

**Returns:**
- The cosine of the input value.

```python
float mDegToRad (float degrees )
```

Convert specified degrees into radians.

**Parameters:**
- \( degrees \) Input Value (in degrees).

**Returns:**
- The specified degrees value converted to radians.

```python
string mFloatLength (float v,
                     int   precision
)
```

Formats the specified number to the given number of decimal places.

**Parameters:**
- \( v \) Number to format.
- \( precision \) Number of decimal places to format to (1-9).

**Returns:**
- Number formatted to the specified number of decimal places.
int mFloor(float v )

Round v down to the nearest integer.

**Parameters:**
- v Number to convert to integer.

**Returns:**
- Number converted to integer.

float mFMod(float v, float d )

Calculate the remainder of v/d.

**Parameters:**
- v Input Value.
- d Divisor Value.

**Returns:**
- The remainder of v/d.

bool mIsPow2(int v )

Returns whether the value is an exact power of two.

**Parameters:**
- v Input value.

**Returns:**
- Whether the specified value is an exact power of two.
float mLerp(float v1,  
    float v2,  
    float time  
)

Calculate linearly interpolated value between two specified numbers using specified normalized time.

Parameters:
- \(v1\) Interpolate From Input value.
- \(v2\) Interpolate To Input value.
- \(time\) Normalized time used to interpolate values (0-1).

Returns:
The interpolated value between the two specified values at normalized time \(t\).

float mLog(float v)

Calculate the natural logarithm of \(v\).

Parameters:
- \(v\) Input Value.

Returns:
The natural logarithm of the input value.

float mPi()

Return the value of PI (half-circle in radians).

Returns:
The value of PI.
**float mPow(float v, float p)**

Calculate b raised to the p-th power.

**Parameters:**
- \(v\) Input Value.
- \(p\) Power to raise value by.

**Returns:**
- \(v\) raised to the p-th power.

**float mRadToDeg(float radians)**

Convert specified radians into degrees.

**Parameters:**
- \(radians\) Input Value (in radians).

**Returns:**
- The specified radians value converted to degrees.

**int mRound(float v)**

Round \(v\) to the nearest integer.

**Parameters:**
- \(v\) Number to convert to integer.

**Returns:**
- Number converted to integer.
float mSaturate (float v )

Clamp the specified value between 0 and 1 (inclusive).

**Parameters:**

v Input value.

**Returns:**

The specified value clamped between 0 and 1 (inclusive).

float mSin(float v )

Calculate the sine of v.

**Parameters:**

v Input Value (in radians).

**Returns:**

The sine of the input value.

string mSolveCubic (float a,
     float b,
     float c,
     float d
     )

Solve a cubic equation (3rd degree polynomial) of form a*x^3 + b*x^2 + c*x + d = 0.

**Parameters:**

a First Coefficient.

b Second Coefficient.

c Third Coefficient.
\[ d \text{ Fourth Coefficient.} \]

**Returns:**
A 4-tuple, containing: (sol x0 x1 x2). (sol) is the number of solutions (being 0, 1, 2 or 3), and (x0), (x1) and (x2) are the solutions, if any.

```c
string mSolveQuadratic(float a,
                        float b,
                        float c
)
```

Solve a quadratic equation (2nd degree polynomial) of form \( ax^2 + bx + c = 0 \).

**Parameters:**
- \( a \) First Coefficient.
- \( b \) Second Coefficient.
- \( c \) Third Coefficient.

**Returns:**
A triple, containing: (sol x0 x1). (sol) is the number of solutions (being 0, 1, or 2), and (x0) and (x1) are the solutions, if any.

```c
string mSolveQuartic(float a,
                      float b,
                      float c,
                      float d,
                      float e
)
```

Solve a quartic equation (4th degree polynomial) of form \( ax^4 + bx^3 + cx^2 + dx + e = 0 \).
Parameters:

- **a** First Coefficient.
- **b** Second Coefficient.
- **c** Third Coefficient.
- **d** Fourth Coefficient.
- **e** Fifth Coefficient.

**Returns:**

A 5-tuple, containing: (sol x0 x1 x2 c3). (sol) is the number of solutions (being 0, 1, 2, 3 or 4), and (x0), (x1), (x2) and (x3) are the solutions, if any.

```plaintext
float mSqrt(float v )
```

Calculate the square-root of v.

**Parameters:**

- **v** Input Value.

**Returns:**

The square-root of the input value.

```plaintext
float mTan(float v )
```

Calculate the tangent of v.

**Parameters:**

- **v** Input Value (in radians).

**Returns:**

The tangent of the input value.
Matrix Math
[Math]

Functions for working with matrices (MatrixF, AngAxisF, MatrixRotation, MatrixPosition). More...
## Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformF</td>
<td>MatrixCreate (VectorF position, AngAxisF orientation)</td>
<td>Create a transform from the given translation and orientation.</td>
</tr>
<tr>
<td>TransformF</td>
<td>MatrixCreateFromEuler (Point3F angles)</td>
<td>A matrix from the given rotations.</td>
</tr>
<tr>
<td>Point3F</td>
<td>MatrixMulPoint (TransformF transform, Point3F point)</td>
<td>Multiply the given point by the given transform assuming that w=1.</td>
</tr>
<tr>
<td>TransformF</td>
<td>MatrixMultiply (TransformF left, TransformF right)</td>
<td>Multiply the two matrices.</td>
</tr>
<tr>
<td>VectorF</td>
<td>MatrixMulVector (TransformF transform, VectorF vector)</td>
<td>Multiply the vector by the transform assuming that w=0.</td>
</tr>
</tbody>
</table>
Detailed Description

Functions for working with matrices (MatrixF, AngAxisF, MatrixRotation, MatrixPosition).
Function Documentation

**TransformF MatrixCreate** (VectorF position, AngAxisF orientation)

Create a transform from the given translation and orientation.

**Parameters:**
- *position* The translation vector for the transform.
- *orientation* The axis and rotation that orients the transform.

**Returns:**
A transform based on the given position and orientation.

**TransformF MatrixCreateFromEuler** (Point3F angles)

a matrix from the given rotations.

**Parameters:**
- *Vector3F X, Y, and Z rotation in *radians*.

**Returns:**
A transform based on the given orientation.

**Point3F MatrixMulPoint** (TransformF transform, Point3F point)

Multiply the given point by the given transform assuming that w=1.

This function will multiply the given vector such that translation with take effect.
Parameters:

\( transform \) A transform.
\( point \) A vector.

Returns:
The transformed vector.

\[
\text{TransformF MatrixMultiply (TransformF left, TransformF right)}
\]

Multiply the two matrices.

Parameters:

\( left \) First transform.
\( right \) Right transform.

Returns:
Concatenation of the two transforms.

\[
\text{VectorF MatrixMulVector (TransformF transform, VectorF vector)}
\]

Multiply the vector by the transform assuming that \( w=0 \).

This function will multiply the given vector by the given transform such that translation will not affect the vector.

Parameters:

\( transform \) A transform.
\( vector \) A vector.

Returns:
The transformed vector.
Random Numbers
[Math]

Functions for generating random numbers. More...
## Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>float</code></td>
<td><code>getRandom(int a, int b)</code></td>
<td>Returns a random number based on parameters passed in.</td>
</tr>
<tr>
<td><code>int</code></td>
<td><code>getRandomSeed()</code></td>
<td>Get the current seed used by the random number generator.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>setRandomSeed(int seed=-1)</code></td>
<td>Set the current seed for the random number generator.</td>
</tr>
</tbody>
</table>
Detailed Description

Functions for generating random numbers.

Based on a seed, the random number generator produces a sequence of numbers. As a given seed will always produce the same sequence of numbers this can be used to generate reproducible sequences of apparently random numbers.

To set the seed, call setRandomSeed().
**Function Documentation**

`float getRandom(int a, int b)`

Returns a random number based on parameters passed in.

If no parameters are passed in, `getRandom()` will return a float between 0.0 and 1.0. If one parameter is passed an integer between 0 and the passed in value will be returned. Two parameters will return an integer between the specified numbers.

**Parameters:**

- `a`: If this is the only parameter, a number between 0 and `a` is returned. Elsewise represents the lower bound.
- `b`: Upper bound on the random number. The random number will be <= `b`.

**Returns:**

A pseudo-random integer between `a` and `b`, between 0 and `a`, or a float between 0.0 and 1.1 depending on usage.

**Note:**

All parameters are optional.

**See also:**

- `setRandomSeed`

`int getRandomSeed()`

Get the current seed used by the random number generator.

**Returns:**

The current random number generator seed value.
void setRandomSeed(int seed = -1)

Set the current seed for the random number generator.

Based on this seed, a repeatable sequence of numbers will be produced by `getRandom()`.

**Parameters:**

The seed with which to initialize the random number generator with. The same seed will always lead to the same sequence of pseudo-random numbers. If -1, the current timestamp will be used as the seed which is a good basis for randomization.

Copyright © GarageGames, LLC. All Rights Reserved.
Vector Math
[Math]

Functions for working with three-dimensional vectors (VectorF/Point3F). More...
## Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VectorF</td>
<td>VectorAdd (VectorF a, VectorF b)</td>
<td>Add two vectors.</td>
</tr>
<tr>
<td>VectorF</td>
<td>VectorCross (VectorF a, VectorF b)</td>
<td>Calculate the cross product of two vectors.</td>
</tr>
<tr>
<td>float</td>
<td>VectorDist (VectorF a, VectorF b)</td>
<td>Compute the distance between two vectors.</td>
</tr>
<tr>
<td>float</td>
<td>VectorDot (VectorF a, VectorF b)</td>
<td>Compute the dot product of two vectors.</td>
</tr>
<tr>
<td>float</td>
<td>VectorLen (VectorF v)</td>
<td>Calculate the magnitude of the given vector.</td>
</tr>
<tr>
<td>VectorF</td>
<td>VectorLerp (VectorF a, VectorF b, float t)</td>
<td>Linearly interpolate between two vectors by $t$.</td>
</tr>
<tr>
<td>VectorF</td>
<td>VectorNormalize (VectorF v)</td>
<td>Brings a vector into its unit form, i.e. such that it has the magnitude 1.</td>
</tr>
<tr>
<td>MatrixF</td>
<td>VectorOrthoBasis (AngAxisF aa)</td>
<td>Create an orthogonal basis from the given vector.</td>
</tr>
<tr>
<td>VectorF</td>
<td>VectorScale (VectorF a, float scalar)</td>
<td>Scales a vector by a scalar.</td>
</tr>
<tr>
<td>VectorF</td>
<td>VectorSub (VectorF a, VectorF b)</td>
<td>Subtract two vectors.</td>
</tr>
</tbody>
</table>
Detailed Description

Functions for working with three-dimensional vectors (VectorF/Point3F).
VectorF VectorAdd(VectorF a, VectorF b)

Add two vectors.

Parameters:
   a  The first vector.
   b  The second vector.

Returns:
   The vector $a + b$.

Example:

```c
// VectorAdd( %a, %b );
//
// The sum of vector a, (ax, ay, az), and
//
//     a + b = ( ax + bx, ay + by, az + bz );

%a = "1 0 0";
%b = "0 1 0";

%r = "( 1 + 0, 0 + 1, 0 + 0 )";
%r = "1 1 0";
%r = VectorAdd( %a, %b );
```
VectorF VectorCross(VectorF a, VectorF b)

Calculate the cross product of two vectors.

Parameters:
  a  The first vector.
  b  The second vector.

Returns:
  The cross product \( x \).

Example:

```plaintext
//-------------------------------------------------------------------
//
//  VectorCross( %a, %b );
//
//  The cross product of vector a, (ax, ay, az), and vector b, (bx, by, bz), is
//  \( a \times b = ( ( ay \times bz ) - ( az \times by ), ( az \times bx ) - ( ax \times bz ), ( ax \times by ) - ( ay \times bx ) ) \).
//
//-------------------------------------------------------------------

%a = "1 1 0";
%b = "2 0 1";

// %r = "( ( 1 * 1 ) - ( 0 * 0 ), ( 0 * 2 ) - ( 0 * 2 ) )",
// %r = "1 -1 -2"
%r = VectorCross( %a, %b );
```

float VectorDist(VectorF a, VectorF b)
Compute the distance between two vectors.

**Parameters:**

- $a$ The first vector.
- $b$ The second vector.

**Returns:**

The length($b - a$).

**Example:**

```c
// --------------------------------------------------------------------------------------------------
//
// VectorDist( %a, %b );
//
// The distance between vector a, (ax, ay, //
// a -> b = ||( b - a )||
// = ||( bx - ax, by - ay, bz .
// = mSqrt( ( bx - ax ) * ( bx
//
// --------------------------------------------------------------------------------------------------

%a = "1 1 0";
%b = "2 0 1";

// %r = mSqrt( ( 2 - 1 ) * ( 2 - 1 ) + ( 0 // %r = mSqrt( 3 );
%r = VectorDist( %a, %b );
```

```c
float VectorDot (VectorF a,
    VectorF b
```
Compute the dot product of two vectors.

**Parameters:**

- *a* The first vector.
- *b* The second vector.

**Returns:**

The dot product \( a \cdot b \).

**Example:**

```c
// VectorDot( %a, %b );

// The dot product between vector a, (ax, ay, az),
// a . b = ( ax * bx + ay * by + az * bz )

%a = "1 1 0";
%b = "2 0 1";

%r = "( 1 * 2 + 1 * 0 + 0 * 1 )";
%r = 2;
%r = VectorDot( %a, %b );
```

**float VectorLen(VectorF v )**

Calculate the magnitude of the given vector.
Parameters:

\( v \) A vector.

Returns:

The length of vector \( v \).

Example:

```c
// VectorLen( %a );
//
// The length or magnitude of vector \( a \), (ax, ay, az), is:
//
// \[ ||a|| = \sqrt{ ax \times ax + ay \times ay + az \times az } \]

%a = "1 1 0";

// %r = mSqrt( 1 \times 1 + 1 \times 1 + 0 \times 0 );
// %r = mSqrt( 2 );
// %r = 1.414;
%r = VectorLen( %a );
```

```c
VectorF VectorLerp(VectorF a, 
                    VectorF b, 
                    float t 
)
```

Linearly interpolate between two vectors by \( t \).

Parameters:

\( a \) Vector to start interpolation from.
Vector to interpolate to.
Interpolation factor (0-1). At zero, \( a \) is returned and at one, \( b \) is returned. In between, an interpolated vector between \( a \) and \( b \) is returned.

**Returns:**
An interpolated vector between \( a \) and \( b \).

**Example:**

```plaintext
// VectorLerp( %a, %b );

// The point between vector a, (ax, ay, az),
// weighted by the interpolation factor, t
//
// r = a + t * ( b - a )
// = ( ax + t * ( bx - ax ), ay + t
//

%a = "1 1 0";
%b = "2 0 1";
%v = "0.25";

%r = VectorLerp( %a, %b );
```

**VectorF VectorNormalize(VectorF v)**

Brings a vector into its unit form, i.e. such that it has the magnitude 1.
Parameters:

\( v \) The vector to normalize.

Returns:

The vector \( v \) scaled to length 1.

Example:

```cpp
// VectorNormalize( %a );
// The normalized vector a, (ax, ay, az),
// a^ = a / ||a||
// = ( ax / ||a||, ay / ||a||, az ),
// %a = "1 1 0";
%1 = 1.414;
// %r = "( 1 / 1.141, 1 / 1.141, 0 / 1.141 )";
// %r = "0.707 0.707 0";
%r = VectorNormalize( %a );
```

MatrixF VectorOrthoBasis(AngAxisF aa)

Create an orthogonal basis from the given vector.

Parameters:

\( aaf \) The vector to create the orthogonal basis from.

Returns:
A matrix representing the orthogonal basis.

VectorF VectorScale (VectorF a, 
    float scalar
)

Scales a vector by a scalar.

Parameters:
  a       The vector to scale.
  scalar The scale factor.

Returns:
  The vector \( a \times \text{scalar} \).

Example:

```cpp
// VectorScale( %a, %v );
// Scaling vector a, (ax, ay, az), but the
// a * v = ( ax * v, ay * v, az * v )

%a = "1 1 0";
%v = "2";

%r = VectorScale( %a, %v );
```
VectorF VectorSub(VectorF a, VectorF b)

Subtract two vectors.

Parameters:
    a The first vector.
    b The second vector.

Returns:
    The vector a - b.

Example:

    // VectorSub( %a, %b );
    // The difference of vector a, (ax, ay, az);
    //     a - b = ( ax - bx, ay - by, az - bz );
    //
    %a = "1 0 0";
    %b = "0 1 0";
    // %r = "( 1 - 0, 0 - 1, 0 - 0 )";
    // %r = "1 -1 0";
    %r = VectorSub( %a, %b );
Strings
[Core]

Functions for dealing with string values. More...
# Modules

## Field Manipulators

Functions to deal with whitespace-separated lists of values in strings.
**Functions**

<table>
<thead>
<tr>
<th>Description</th>
<th>Signature</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replace all escape sequences in text with their respective character codes.</td>
<td>string collapseEscape (string text)</td>
<td>Replace all escape sequences in text with their respective character codes.</td>
</tr>
<tr>
<td>Dumps information about String memory usage.</td>
<td>void dumpStringMemStats ()</td>
<td>Dumps information about String memory usage.</td>
</tr>
<tr>
<td>Test whether the given string ends with the given suffix.</td>
<td>bool endsWith (string str, string suffix, bool caseSensitive=false)</td>
<td>Test whether the given string ends with the given suffix.</td>
</tr>
<tr>
<td>Replace all characters in text that need to be escaped for the string to be a valid string literal with their respective escape sequences.</td>
<td>string expandEscape (string text)</td>
<td>Replace all characters in text that need to be escaped for the string to be a valid string literal with their respective escape sequences.</td>
</tr>
<tr>
<td>Return a substring of str starting at start and continuing either through to the end of str (if numChars is -1) or for numChars characters (except if this would exceed the actual source string length).</td>
<td>string getSubStr (string str, int start, int numChars=-1)</td>
<td>Return a substring of str starting at start and continuing either through to the end of str (if numChars is -1) or for numChars characters (except if this would exceed the actual source string length).</td>
</tr>
<tr>
<td>Get the numeric suffix of the given input string.</td>
<td>int getTrailingNumber (string str)</td>
<td>Get the numeric suffix of the given input string.</td>
</tr>
<tr>
<td>Test whether the character at the given position is an alphanumeric character.</td>
<td>bool isalnum (string str, int index)</td>
<td>Test whether the character at the given position is an alphanumeric character.</td>
</tr>
<tr>
<td>Test whether the character at the given position is a whitespace character.</td>
<td>bool isspace (string str, int index)</td>
<td>Test whether the character at the given position is a whitespace character.</td>
</tr>
<tr>
<td>Remove leading whitespace from the string.</td>
<td>string ltrim (string str)</td>
<td>Remove leading whitespace from the string.</td>
</tr>
<tr>
<td>Tokenize a string using a set of delimiting characters.</td>
<td>string nextToken (string str, string token, string delimiters)</td>
<td>Tokenize a string using a set of delimiting characters.</td>
</tr>
<tr>
<td>Remove trailing whitespace from the string.</td>
<td>string rtrim (string str)</td>
<td>Remove trailing whitespace from the string.</td>
</tr>
<tr>
<td>Function</td>
<td>Signature</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>bool</td>
<td><code>startsWith (string str, string prefix, bool caseSensitive=false)</code></td>
<td>Test whether the given string begins with the given prefix.</td>
</tr>
<tr>
<td>int</td>
<td><code>strasc (string chr)</code></td>
<td>Return the integer character code value corresponding to the first character in the given string.</td>
</tr>
<tr>
<td>string</td>
<td><code>strchr (string str, string chr)</code></td>
<td>Find the first occurrence of the given character in <code>str</code>.</td>
</tr>
<tr>
<td>int</td>
<td><code>strchrpos (string str, string chr, int start=0)</code></td>
<td>Find the first occurrence of the given character in the given string.</td>
</tr>
<tr>
<td>int</td>
<td><code>strcmp (string str1, string str2)</code></td>
<td>Compares two strings using case-sensitive comparison.</td>
</tr>
<tr>
<td>string</td>
<td><code>strformat (string format, string value)</code></td>
<td>Format the given value as a string using printf-style formatting.</td>
</tr>
<tr>
<td>int</td>
<td><code>stricmp (string str1, string str2)</code></td>
<td>Compares two strings using case-insensitive comparison.</td>
</tr>
<tr>
<td>int</td>
<td><code>strnatcmp (string str1, string str2)</code></td>
<td>Compares two strings using &quot;natural order&quot; case-insensitive comparison.</td>
</tr>
<tr>
<td>string</td>
<td><code>stripChars (string str, string chars)</code></td>
<td>Remove all occurrences of characters contained in <code>chars</code> from <code>str</code>.</td>
</tr>
<tr>
<td>String</td>
<td><code>stripTrailingNumber (string str)</code></td>
<td>Strip a numeric suffix from the given string.</td>
</tr>
<tr>
<td>bool</td>
<td><code>strIsMatchExpr (string pattern, string str, bool caseSensitive=false)</code></td>
<td>Match a pattern against a string.</td>
</tr>
<tr>
<td>bool</td>
<td><code>strIsMatchMultipleExpr (string patterns, string str, bool caseSensitive=false)</code></td>
<td>Match a multiple patterns against a single string.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><code>int strlen (string str)</code></td>
<td>Get the length of the given string in bytes.</td>
<td></td>
</tr>
<tr>
<td><code>string strlwr (string str)</code></td>
<td>Return an all lower-case version of the given string.</td>
<td></td>
</tr>
<tr>
<td><code>int strnatcmp (string str1, string str2)</code></td>
<td>Compares two strings using &quot;natural order&quot; case-sensitive comparison.</td>
<td></td>
</tr>
<tr>
<td><code>int strpos (string haystack, string needle, int offset=0)</code></td>
<td>Find the start of <code>needle</code> in <code>haystack</code> searching from left to right beginning at the given offset.</td>
<td></td>
</tr>
<tr>
<td><code>string strrchr (string str, string chr)</code></td>
<td>Find the last occurrence of the given character in <code>str</code>.</td>
<td></td>
</tr>
<tr>
<td><code>int strrchrpos (string str, string chr, int start=0)</code></td>
<td>Find the last occurrence of the given character in the given string.</td>
<td></td>
</tr>
<tr>
<td><code>string strrepeat (string str, int numTimes, string delimiter=&quot;&quot;)</code></td>
<td>Return a string that repeats <code>str</code> <code>numTimes</code> number of times delimiting each occurrence with <code>delimiter</code>.</td>
<td></td>
</tr>
<tr>
<td><code>string strreplace (string source, string from, string to)</code></td>
<td>Replace all occurrences of <code>from</code> in <code>source</code> with <code>to</code>.</td>
<td></td>
</tr>
<tr>
<td><code>intstrstr (string string, string substring)</code></td>
<td>Find the start of <code>substring</code> in the given <code>string</code> searching from left to right.</td>
<td></td>
</tr>
<tr>
<td><code>string strupr (string str)</code></td>
<td>Return an all upper-case version of the given string.</td>
<td></td>
</tr>
<tr>
<td><code>string trim (string str)</code></td>
<td>Remove leading and trailing whitespace from the string.</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

Functions for dealing with string values.

Since in TorqueScript any value is implicitly also a string, these functions can be used with all values.
Function Documentation

string collapseEscape(string text )

Replace all escape sequences in text with their respective character codes.

This function replaces all escape sequences (\n, \t, etc) in the given string with the respective characters they represent.

The primary use of this function is for converting strings from their literal form into their compiled/translated form, as is normally done by the TorqueScript compiler.

Parameters:

  text  A string.

Returns:

  A duplicate of text with all escape sequences replaced by their respective character codes.

Example:

```
// Print:
//
//    str
//    ing
//
// to the console. Note how the backslash
// in order to prevent the TorqueScript co
// sequence in the resulting string.
echo( collapseEscape( "str\ning" ) );
```

See also:

  expandEscape
void dumpStringMemStats( )

Dumps information about String memory usage.

bool endsWith( string str,
                string suffix,
                bool caseSensitive = false )

Test whether the given string ends with the given suffix.

Parameters:

- **str**: The string to test.
- **suffix**: The potential suffix of **str**.
- **caseSensitive**: If true, the comparison will be case-sensitive; if false, differences in casing will not be taken into account.

Returns:

- True if the last characters in **str** match the complete contents of **suffix**; false otherwise.

Example:

```
startsWith( "TEST123", "123" ) // Returns
```

See also:

- **startsWith**

string expandEscape( string text )

Replace all characters in **text** that need to be escaped for the string to be a valid string literal with their respective escape sequences.
All characters in text that cannot appear in a string literal will be replaced by an escape sequence (\n, \t, etc).

The primary use of this function is for converting strings suitable for being passed as string literals to the TorqueScript compiler.

**Parameters:**

- **text** A string

**Returns:**

A duplicate of the text parameter with all unescaped characters that cannot appear in string literals replaced by their respective escape sequences.

```plaintext
expandEscape( "str" NL "ing" ) // Returns "str\ning".
```

**See also:**

- collapseEscape

---

```plaintext
string getSubStr (string str, int start, int numChars = -1)
```

Return a substring of str starting at start and continuing either through to the end of str (if numChars is -1) or for numChars characters (except if this would exceed the actual source string length).

**Parameters:**

- **str** The string from which to extract a substring.
- **start** The offset at which to start copying out characters.
- **numChars** Optional argument to specify the number of characters to copy. If this is -1, all characters up the end of the input string are copied.
Returns:
A string that contains the given portion of the input string.

Example:
```
getSubStr( "foobar", 1, 2 ) // Returns "oo"
```

```cpp
int getTrailingNumber(string str)
```

Get the numeric suffix of the given input string.

**Parameters:**

- `str` The string from which to read out the numeric suffix.

**Returns:**

The numeric value of the number suffix of `str` or -1 if `str` has no such suffix.

**Example:**
```
getTrailingNumber( "test123" ) // Returns 123
```

**See also:**

`stripTrailingNumber`

```cpp
bool isalnum(string str, int index)
```

Test whether the character at the given position is an alpha-numeric character.

Alpha-numeric characters are characters that are either alphabetic (a-z, A-Z) or numbers (0-9).
Parameters:

\( str \)  The string to test.
\( index \)  The index of a character in \( str \).

Returns:

True if the character at the given index in \( str \) is an alpha-numeric character; false otherwise.

See also:

isspace

```c
bool isspace (string str,
            int index)
```

Test whether the character at the given position is a whitespace character.

Characters such as tab, space, or newline are considered whitespace.

Parameters:

\( str \)  The string to test.
\( index \)  The index of a character in \( str \).

Returns:

True if the character at the given index in \( str \) is a whitespace character; false otherwise.

See also:

isalnum

```c
string ltrim (string str)
```
Remove leading whitespace from the string.

**Parameters:**

str A string.

**Returns:**

A string that is the same as str but with any leading (i.e. leftmost) whitespace removed.

**Example:**

```plaintext
ltrim( " string " ); // Returns "string"
```

**See also:**

rtrim
trim

```plaintext
string nextToken(string str,
    string token,
    string delimiters
)
```

Tokenize a string using a set of delimiting characters.

This function first skips all leading characters in str that are contained in `delimiters`. From that position, it then scans for the next character in str that is contained in `delimiters` and stores all characters from the starting position up to the first delimiter in a variable in the current scope called `token`. Finally, it skips all characters in `delimiters` after the token and then returns the remaining string contents in `str`.

To scan out all tokens in a string, call this function repeatedly by passing the result it returns each time as the new `str` until the function returns `""`. 
Parameters:

- **str**: A string.
  - The name of the variable in which to store the current token. This variable is set in the scope in which nextToken is called.
- **token**: The name of the variable in which to store the current token. This variable is set in the scope in which nextToken is called.
- **delimiters**: A string of characters. Each character is considered a delimiter.

Returns:

The remainder of str after the token has been parsed out or "" if no more tokens were found in str.

Example:

```c
// Prints:
// a
// b
// c
%str = "a  b  c";
while ( %str !$= "" )
{
    // First time, stores "a" in the variable
    %str = nextToken( %str, "token", " " );
    echo( %token );
}
```

**string rtrim(string str )**

Remove trailing whitespace from the string.

Parameters:

- **str**: A string.

Returns:

A string that is the same as str but with any trailing (i.e.
rightmost) whitespace removed.

Example:
```c
rtrim( " string " ); // Returns " str
```

See also:
- ltrim
- trim

```c
bool startsWith(string str,
               string prefix,
               bool caseSensitive = false
)
```

Test whether the given string begins with the given prefix.

**Parameters:**
- `str` The string to test.
- `prefix` The potential prefix of `str`. If true, the comparison will be case-sensitive; if `caseSensitive` false, differences in casing will not be taken into account.

**Returns:**
True if the first characters in `str` match the complete contents of `prefix`; false otherwise.

Example:
```c
startsWith( "TEST123", "test" ) // Returns
```

See also:
- `endsWith`
Return the integer character code value corresponding to the first character in the given string.

**Parameters:**

`chr` a (one-character) string.

**Returns:**

the UTF32 code value for the first character in the given string.

Find the first occurrence of the given character in `str`.

**Parameters:**

`str` The string to search.

`chr` The character to search for. Only the first character from the string is taken.

**Returns:**

The remainder of the input string starting with the given character or the empty string if the character could not be found.

**See also:**

`strrchr`

The remainder of the input string starting with the given character or the empty string if the character could not be found.

**See also:**

`strrchr`
Find the first occurrence of the given character in the given string.

**Parameters:**
- `str` The string to search.
- `chr` The character to look for. Only the first character of this string will be searched for.
- `start` The index into `str` at which to start searching for the given character.

**Returns:**
The index of the first occurrence of `chr` in `str` or `-1` if `str` does not contain the given character.

**Example:**
```
strchrpos( "test", "s" ) // Returns 2.
```

**int strcmp(string str1, string str2)**

Compares two strings using case-**sensitive** comparison.

**Parameters:**
- `str1` The first string.
- `str2` The second string.

**Returns:**
0 if both strings are equal, a value `<0` if the first character different in `str1` has a smaller character code value than the character at the same position in `str2`, and a value `>1` otherwise.

**Example:**
if( strcmp( %var, "foobar" ) == 0 )
    echo( "%var is equal to 'foobar'" );

See also:
    strcmp
    strnatcmp

string strformat( string format,
                  string value
 )

Format the given value as a string using printf-style formatting.

Parameters:
    format  A printf-style format string.
    value   The value argument matching the given format string.

Example:

    // Convert the given integer value to a string.
    %hex = strformat( "%x", %value );

See also:
    http://en.wikipedia.org/wiki/Printf

int strcmp( string str1,
            string str2
 )

Compares two strings using case-**insensitive** comparison.

Parameters:
    str1  The first string.
    str2  The second string.
Returns:

0 if both strings are equal, a value <0 if the first character different in str1 has a smaller character code value than the character at the same position in str2, and a value >0 otherwise.

Example:

```c
if(stricmp("FOObar", "foobar") == 0)
    echo("this is always true");
```

See also:

- strcmp
- strnatcmp

```c
int strnatcmp(string str1,
               string str2
)
```

Compares two strings using "natural order" case-insensitive comparison.

Natural order means that rather than solely comparing single character code values, strings are ordered in a natural way. For example, the string "hello10" is considered greater than the string "hello2" even though the first numeric character in "hello10" actually has a smaller character value than the corresponding character in "hello2". However, since 10 is greater than 2, strnatcmp will put "hello10" after "hello2".

Parameters:

- `str1` The first string.
- `str2` The second string.

Returns:

0 if the strings are equal, a value >0 if `str1` comes after `str2` in
a natural order, and a value <0 if str1 comes before str2 in a natural order.

Example:

// Bubble sort 10 elements of %array using
do
{
    %swapped = false;
    for( %i = 0; %i < 10 - 1; %i ++ )
        if( strnatcmp( %array[ %i ], %array[ %i + 1 ] ) > 0 )
        {
            %temp = %array[ %i ];
            %array[ %i ] = %array[ %i + 1 ];
            %array[ %i + 1 ] = %temp;
            %swapped = true;
        }
}
while( %swapped );

See also:
    stricmp
    strnatcmp

string stripChars(string str, string chars )

Remove all occurrences of characters contained in chars from str.

Parameters:
    str    The string to filter characters out from.
    chars  A string of characters to filter out from str.

Returns:
A version of \textit{str} with all occurrences of characters contained in \textit{chars} filtered out.

**Example:**

\begin{verbatim}
stripChars( "teststring", "se" );  // Returns "tttring".
\end{verbatim}

\begin{verbatim}
String stripTrailingNumber( string str )
\end{verbatim}

Strip a numeric suffix from the given string.

**Parameters:**

\textit{str} The string from which to strip its numeric suffix.

**Returns:**

The string \textit{str} without its number suffix or the original string \textit{str} if it has no such suffix.

**Example:**

\begin{verbatim}
stripTrailingNumber( "test123" )  // Returns "test".
\end{verbatim}

See also:

getTrailingNumber

\begin{verbatim}
bool strIsMatchExpr( string pattern,
    string str,
    bool caseSensitive = false
  )
\end{verbatim}

Match a pattern against a string.

**Parameters:**

The wildcard pattern to match against. The pattern can include characters, '*' to match any
pattern number of characters and '?' to match a single character.

str The string which should be matched against pattern.
If true, characters in the pattern are matched caseSensitive in case-sensitive fashion against this string. If false, differences in casing are ignored.

Returns:
True if str matches the given pattern.

Example:
```cpp
strIsMatchExpr( "f?o*R", "foobar" ) // True
```

See also:
strIsMatchMultipleExpr

```cpp
bool strIsMatchMultipleExpr (string patterns,
    string str,
    bool caseSensitive = false
)
```

Match a multiple patterns against a single string.

Parameters:

- **patterns** A tab-separated list of patterns. Each pattern can include characters, '*' to match any number of characters and '?' to match a single character. Each of the patterns is tried in turn.
- **str** The string which should be matched against patterns.
- **caseSensitive** If true, characters in the pattern are matched in case-sensitive fashion against this string. If false, differences in casing are ignored.
**Returns:**
True if `str` matches any of the given *patterns*.

**Example:**
```c
strIsMatchMultipleExpr( "*.cs *.gui *.mis"
```

**See also:**
strIsMatchExpr

```c
int strlen(string str )
```

Get the length of the given string in bytes.

**Note:**
This does **not** return a true character count for strings with multi-byte characters!

**Parameters:**
`str` A string.

**Returns:**
The length of the given string in bytes.

```c
string strlwr(string str )
```

Return an all lower-case version of the given string.

**Parameters:**
`str` A string.

**Returns:**
A version of `str` with all characters converted to lower-case.
Example:

```c
strlwr( "TesT1" ) // Returns "test1"
```

See also:

strupr

```c
int strnatcmp(string str1,
               string str2)
```

Compares two strings using "natural order" case-sensitive comparison.

Natural order means that rather than solely comparing single character code values, strings are ordered in a natural way. For example, the string "hello10" is considered greater than the string "hello2" even though the first numeric character in "hello10" actually has a smaller character value than the corresponding character in "hello2". However, since 10 is greater than 2, strnatcmp will put "hello10" after "hello2".

Parameters:

- `str1` The first string.
- `str2` The second string.

Returns:

- 0 if the strings are equal, a value >0 if `str1` comes after `str2` in a natural order, and a value <0 if `str1` comes before `str2` in a natural order.

Example:

```c
// Bubble sort 10 elements of %array using
do
do {
```
\%swapped = false;
for( \%i = 0; \%i < 10 - 1; \%i ++ )
    if( strnatcmp( \%array[ \%i ], \%array|

    { %temp = \%array[ \%i ];
      \%array[ \%i ] = \%array[ \%i + 1 ];
      \%array[ \%i + 1 ] = \%temp;
      \%swapped = true;
    }
}
while( \%swapped );

See also:

\texttt{strcmp}
\texttt{strnatcmp}

\texttt{int \texttt{strpos(} string \texttt{haystack, string \texttt{needle, int \texttt{offset = 0 \texttt{)}})}}

Find the start of \texttt{needle} in \texttt{haystack} searching from left to right beginning at the given offset.

\textbf{Parameters:}

- \texttt{haystack} The string to search.
- \texttt{needle} The string to search for.

\textbf{Returns:}

The index at which the first occurrence of \texttt{needle} was found in \texttt{haystack} or -1 if no match was found.

\textbf{Example:}

\texttt{strpos( "b ab", "b", 1 ) // Returns 3.}
string strrchr (string str,
        string chr
    )

Find the last occurrence of the given character in \textit{str}.

**Parameters:**

\textit{str}  The string to search.
\textit{chr}  The character to search for. Only the first character from the string is taken.

**Returns:**

The remainder of the input string starting with the given character or the empty string if the character could not be found.

**See also:**

strchr

int strrchrpos (string str,
        string chr,
        int     start = 0
    )

Find the last occurrence of the given character in the given string.

**Parameters:**

\textit{str}  The string to search.
\textit{chr}  The character to look for. Only the first character of this string will be searched for.
\textit{start}  The index into \textit{str} at which to start searching for the given character.
Returns:
The index of the last occurrence of *chr* in *str* or -1 if *str* does not contain the given character.

Example:

```
strrchrpos( "test", "t" ) // Returns 3.
```

```java
string strrepeat(string str,
    int numTimes,
    string delimiter = ""
)
```

Return a string that repeats *str* *numTimes* number of times delimiting each occurrence with *delimiter*.

Parameters:
- `str` The string to repeat multiple times.
- `numTimes` The number of times to repeat `str` in the result string.
- `delimiter` The string to put between each repetition of `str`.

Returns:
A string containing *str* repeated *numTimes* times.

Example:

```
strrepeat( "a", 5, "b" ) // Returns "ababa"
```

```java
string strreplace(string source,
    string from,
    string to
)
```

Replace all occurrences of *from* in *source* with *to*.
Parameters:

- **source**: The string in which to replace the occurrences of *from*.
- **from**: The string to replace in *source*.
- **to**: The string with which to replace occurrences of.

**Returns:**

A string with all occurrences of *from* in *source* replaced by *to*.

**Example:**

```
strreplace( "aabbccbb", "bb", "ee" ) // Returns "aaeeccee".
```

```c
int strstr( string string, string substring )
```

Find the start of *substring* in the given *string* searching from left to right.

**Parameters:**

- **string**: The string to search.
- **substring**: The string to search for.

**Returns:**

The index into *string* at which the first occurrence of *substring* was found or -1 if *substring* could not be found.

**Example:**

```
strstr( "abcd", "c" ) // Returns 2.
```

```c
string strupr( string str )
```

Return an all upper-case version of the given string.
Parameters:

str A string.

Returns:

A version of str with all characters converted to upper-case.

Example:

```cpp
strupr( "TesT1" ) // Returns "TEST1"
```

See also:

strlwr

string trim(string str )

Remove leading and trailing whitespace from the string.

Parameters:

str A string.

Returns:

A string that is the same as str but with any leading (i.e. leftmost) and trailing (i.e. rightmost) whitespace removed.

Example:

```cpp
trim( " string " ); // Returns "string"
```
Field Manipulators
[Strings]

Functions to deal with whitespace-separated lists of values in strings. More...
## Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Signature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>firstWord</code></td>
<td><code>string firstWord (string text)</code></td>
<td>Return the first word in <code>text</code>.</td>
</tr>
<tr>
<td><code>getField</code></td>
<td><code>string getField (string text, int index)</code></td>
<td>Extract the field at the given <code>index</code> in the newline and/or tab separated list in <code>text</code>.</td>
</tr>
<tr>
<td><code>getFieldCount</code></td>
<td><code>int getFieldCount (string text)</code></td>
<td>Return the number of newline and/or tab separated fields in <code>text</code>.</td>
</tr>
<tr>
<td><code>getFields</code></td>
<td><code>string getFields (string text, int startIndex, int endIndex=-1)</code></td>
<td>Extract a range of fields from the given <code>startIndex</code> onwards thru <code>endIndex</code>.</td>
</tr>
<tr>
<td><code>getRecord</code></td>
<td><code>string getRecord (string text, int index)</code></td>
<td>Extract the record at the given <code>index</code> in the newline-separated list in <code>text</code>.</td>
</tr>
<tr>
<td><code>getRecordCount</code></td>
<td><code>int getRecordCount (string text)</code></td>
<td>Return the number of newline-separated records in <code>text</code>.</td>
</tr>
<tr>
<td><code>getRecords</code></td>
<td><code>string getRecords (string text, int startIndex, int endIndex=-1)</code></td>
<td>Extract a range of records from the given <code>startIndex</code> onwards thru <code>endIndex</code>.</td>
</tr>
<tr>
<td><code>getWord</code></td>
<td><code>string getWord (string text, int index)</code></td>
<td>Extract the word at the given <code>index</code> in the whitespace-separated list in <code>text</code>.</td>
</tr>
<tr>
<td><code>getWordCount</code></td>
<td><code>int getWordCount (string text)</code></td>
<td>Return the number of whitespace-separated words in <code>text</code>.</td>
</tr>
<tr>
<td><code>getWords</code></td>
<td><code>string getWords (string text, int startIndex, int endIndex=-1)</code></td>
<td>Extract a range of words from the given <code>startIndex</code> onwards thru <code>endIndex</code>.</td>
</tr>
<tr>
<td><code>removeField</code></td>
<td><code>string removeField (string text, int index)</code></td>
<td>Remove the field in <code>text</code> at the given <code>index</code>.</td>
</tr>
<tr>
<td><code>removeRecord</code></td>
<td><code>string removeRecord (string text, int index)</code></td>
<td>Remove the record in <code>text</code> at the given <code>index</code>.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>removeWord</td>
<td>Remove the word in <code>text</code> at the given <code>index</code>.</td>
<td></td>
</tr>
<tr>
<td>restWords</td>
<td>Return all but the first word in <code>text</code>.</td>
<td></td>
</tr>
<tr>
<td>setField</td>
<td>Replace the field in <code>text</code> at the given <code>index</code> with <code>replacement</code>.</td>
<td></td>
</tr>
<tr>
<td>setRecord</td>
<td>Replace the record in <code>text</code> at the given <code>index</code> with <code>replacement</code>.</td>
<td></td>
</tr>
<tr>
<td>setWord</td>
<td>Replace the word in <code>text</code> at the given <code>index</code> with <code>replacement</code>.</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

Functions to deal with whitespace-separated lists of values in strings.

TorqueScript extensively uses strings to represent lists of values. The functions in this group simplify working with these lists and allow to easily extract individual values from their strings.

The list strings are segregated into three groups according to the delimiters used to separate individual values in the strings:

- Strings of **words**: Elements are separated by newlines (\n), spaces, or tabs (\t).
- Strings of **fields**: Elements are separated by newlines (\n) or tabs (\t).
- Strings of **records**: Elements are separated by newlines (\n).

Aside from the functions here, another useful means to work with strings of words is TorqueScript’s **foreach$** statement.

**foreach$ Statement**
Function Documentation

string firstWord(string text)

Return the first word in text.

Parameters:

  text  A list of words separated by newlines, spaces, and/or tabs.

Returns:

  The word at index 0 in text or "" if text is empty.

Note:

  This is equal to

    getWord( text, 0 )

See also:

  getWord

string getField(string text, int index)

Extract the field at the given index in the newline and/or tab separated list in text.

Fields in text must be separated by newlines and/or tabs.

Parameters:

  text  A list of fields separated by newlines and/or tabs.
  index  The zero-based index of the field to extract.

Returns:
The field at the given index or "" if the index is out of range.

Example:

```c
getField( "a b" TAB "c d" TAB "e f", 1 )
```

See also:
- `getFields`
- `getFieldCount`
- `getWord`
- `getRecord`

```c
int getFieldCount( string text )
```

Return the number of newline and/or tab separated fields in `text`.

**Parameters:**
- `text` A list of fields separated by newlines and/or tabs.

**Returns:**
The number of newline and/or tab sepearated elements in `text`.

Example:

```c
getFieldCount( "a b" TAB "c d" TAB "e f" )
```

See also:
- `getWordCount`
- `getRecordCount`

```c
string getFields( string text,
                 int startIdx,
                 int endIndex = -1
)
```
Extract a range of fields from the given `startIndex` onwards thru `endIndex`.

Fields in `text` must be separated by newlines and/or tabs.

**Parameters:**
- `text` A list of fields separated by newlines and/or tabs.
- `startIndex` The zero-based index of the first field to extract from `text`.
- `endIndex` The zero-based index of the last field to extract from `text`. If this is -1, all fields beginning with `startIndex` are extracted from `text`.

**Returns:**
A string containing the specified range of fields from `text` or "" if `startIndex` is out of range or greater than `endIndex`.

**Example:**
```
getFields( "a b" TAB "c d" TAB "e f", 1 )
```

**See also:**
- `getField`
- `getFieldCount`
- `getWords`
- `getRecords`

```c
string getRecord(string text, int index)
```

Extract the record at the given `index` in the newline-separated list in `text`.

Records in `text` must be separated by newlines.

**Parameters:**
text    A list of records separated by newlines.
index  The zero-based index of the record to extract.

Returns:
The record at the given index or "" if index is out of range.

Example:
getRecord( "a b" NL "c d" NL "e f", 1 ) 

See also:
getRecords
getRecordCount
getWord
getField

int getRecordCount(string text )

Return the number of newline-separated records in text.

Parameters:
    text    A list of records separated by newlines.

Returns:
The number of newline-seperated elements in text.

Example:
getRecordCount( "a b" NL "c d" NL "e f" )

See also:
getWordCount
g getFieldCount

string getRecords(string text,
int startIndex, int endIndex = -1
)

Extract a range of records from the given \texttt{startIndex} onwards thru \texttt{endIndex}.

Records in \texttt{text} must be separated by newlines.

**Parameters:**

- \texttt{text} A list of records separated by newlines.
- \texttt{startIndex} The zero-based index of the first record to extract from \texttt{text}.
- \texttt{endIndex} The zero-based index of the last record to extract from \texttt{text}. If this is -1, all records beginning with \texttt{startIndex} are extracted from \texttt{text}.

**Returns:**

A string containing the specified range of records from \texttt{text} or "" if \texttt{startIndex} is out of range or greater than \texttt{endIndex}.

**Example:**

\texttt{getRecords( "a b" NL "c d" NL "e f", 1 )},

**See also:**

- getRecord
- getRecordCount
- getWords
- getFields

```
string getWord(string text,
    int   index
)
```

Extract the word at the given \texttt{index} in the whitespace-separated
list in text.

Words in text must be separated by newlines, spaces, and/or tabs.

**Parameters:**
- *text*: A whitespace-separated list of words.
- *index*: The zero-based index of the word to extract.

**Returns:**
The word at the given index or "" if the index is out of range.

**Example:**
```
getWord( "a b c", 1 ) // Returns "b"
```

**See also:**
- getWords
- getWordCount
- getField
- getRecord

**int getWordCount(string text)**

Return the number of whitespace-separated words in text.

Words in text must be separated by newlines, spaces, and/or tabs.

**Parameters:**
- *text*: A whitespace-separated list of words.

**Returns:**
The number of whitespace-separated words in text.

**Example:**
```
getWordCount( "a b c d e" ) // Returns 5
```
string getWords(string text,  
    int startIndex,  
    int endIndex = -1  
)  

Extract a range of words from the given *startIndex* onwards thru *endIndex*.

Words in *text* must be separated by newlines, spaces, and/or tabs.

**Parameters:**

- **text**  
  A whitespace-separated list of words.

- **startIndex**  
  The zero-based index of the first word to extract from *text*.

- **endIndex**  
  The zero-based index of the last word to extract from *text*. If this is -1, all words beginning with *startIndex* are extracted from *text*.

**Returns:**

A string containing the specified range of words from *text* or "" if *startIndex* is out of range or greater than *endIndex*.

**Example:**

```csharp
getWords( "a b c d", 1, 2, ) // Returns "bc"
```

**See also:**

getWord  
getWordCount  
getFields  
getFields  
getRecords
string removeField(string text,  
    int index)  

Remove the field in text at the given index.

Fields in text must be separated by newlines and/or tabs.

**Parameters:**
- *text* A list of fields separated by newlines and/or tabs.
- *index* The zero-based index of the field in text.

**Returns:**
A new string with the field at the given index removed or the original string if index is out of range.

**Example:**
```c
removeField("a b" TAB "c d" TAB "e f", 1)
```

**See also:**
- removeWord
- removeRecord

string removeRecord(string text,  
    int index)  

Remove the record in text at the given index.

Records in text must be separated by newlines.

**Parameters:**
- *text* A list of records separated by newlines.
- *index* The zero-based index of the record in text.
**Returns:**
A new string with the record at the given index removed or the original string if index is out of range.

**Example:**
```java
removeRecord( "a b" NL "c d" NL "e f", 1 )
```

**See also:**
removeWord
removeField

```java
string removeWord( string text, int index )
```

Remove the word in text at the given index.

Words in text must be separated by newlines, spaces, and/or tabs.

**Parameters:**
- `text` A whitespace-separated list of words.
- `index` The zero-based index of the word in text.

**Returns:**
A new string with the word at the given index removed or the original string if index is out of range.

**Example:**
```java
removeWord( "a b c d", 2 ) // Returns "a b d"
```

**See also:**
removeField
removeRecord
string restWords(string text)

Return all but the first word in text.

Parameters:

  text A list of words separated by newlines, spaces, and/or tabs.

Returns:

  text with the first word removed.

Note:

  This is equal to
  
  getWords(text, 1)

See also:

  getWords

string setField(string text, int index, string replacement)

Replace the field in text at the given index with replacement.

Fields in text must be separated by newlines and/or tabs.

Parameters:

  text A list of fields separated by newlines and/or tabs.
  index The zero-based index of the field to replace.
  replacement The string with which to replace the field.

Returns:

  A new string with the field at the given index replaced by
replacement or the original string if index is out of range.

Example:

```c
setField("a b" TAB "c d" TAB "e f", 1, "g"
```

See also:

getField
setWord
setRecord

```c
string setRecord(string text, int index, string replacement)
```

Replace the record in text at the given index with replacement.

Records in text must be separated by newlines.

Parameters:

- **text**: A list of records separated by newlines.
- **index**: The zero-based index of the record to replace.
- **replacement**: The string with which to replace the record.

Returns:

A new string with the record at the given index replaced by replacement or the original string if index is out of range.

Example:

```c
setRecord("a b" NL "c d" NL "e f", 1, "g"
```

See also:

getRecord
setWord
setField
string setWord(string text, int index, string replacement)

Replace the word in text at the given index with replacement.

Words in text must be separated by newlines, spaces, and/or tabs.

Parameters:
- text A whitespace-separated list of words.
- index The zero-based index of the word to replace.
- replacement The string with which to replace the word.

Returns:
A new string with the word at the given index replaced by replacement or the original string if index is out of range.

Example:
```c
setWord( "a b c d", 2, "f" ) // Returns "a b f d"
```

See also:
getWord
setField
setRecord
Utilities
[Core]

Miscellaneous utility functions. More...
## Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>int countBits (int v)</code></td>
<td>Count the number of bits that are set in the given 32 bit integer.</td>
</tr>
<tr>
<td>Torque::UUID</td>
<td><code>generateUUID ()</code></td>
</tr>
<tr>
<td></td>
<td>Generate a new universally unique identifier (UUID).</td>
</tr>
</tbody>
</table>
Detailed Description

Miscellaneous utility functions.
Function Documentation

**int countBits ( int v )**

Count the number of bits that are set in the given 32 bit integer.

**Parameters:**

- **v** An integer value.

**Returns:**

- The number of bits that are set in v.

**Torque::UUID generateUUID ( )**

Generate a new universally unique identifier (UUID).

**Returns:**

- A newly generated UUID.
Environment Objects

Objects that represent environmental features, such as, terrain, water, atmosphere, plants and trees. More...
# Modules

<table>
<thead>
<tr>
<th><strong>Water</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Objects that represent water features, from puddles to rivers and oceans.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Atmosphere</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Objects that represent the atmosphere and weather, such as the sky, sun, clouds, and precipitation.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Terrain</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Objects that specialize in representing terrain and other collidable/walkable surfaces.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Forest</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Objects for efficiently placing and rendering trees, rocks, foliage, or any such feature needed in large number.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Foliage</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Objects used for mass replication of foliage, such as grass, rocks, and bushes.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Miscellaneous</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Miscellaneous environmental and level objects.</td>
</tr>
</tbody>
</table>
Detailed Description

Objects that represent environmental features, such as, terrain, water, atmosphere, plants and trees.
Atmosphere
[Environment Objects]

Objects that represent the atmosphere and weather, such as the sky, sun, clouds, and precipitation. More...
# Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BasicClouds</strong></td>
<td>Renders up to three layers of scrolling cloud-cover textures overhead.</td>
</tr>
<tr>
<td><strong>CloudLayer</strong></td>
<td>A layer of clouds which change shape over time and are affected by scene lighting.</td>
</tr>
<tr>
<td><strong>ForestWindEmitter</strong></td>
<td>Object responsible for simulating wind in a level.</td>
</tr>
<tr>
<td><strong>Lightning</strong></td>
<td>An emitter for lightning bolts.</td>
</tr>
<tr>
<td><strong>LightningData</strong></td>
<td>Common data for a Lightning emitter object.</td>
</tr>
<tr>
<td><strong>LightningStrikeEvent</strong></td>
<td>Network event that triggers a lightning strike on the client when it is received.</td>
</tr>
<tr>
<td><strong>Precipitation</strong></td>
<td>Defines a precipitation based storm (rain, snow, etc).</td>
</tr>
<tr>
<td><strong>PrecipitationData</strong></td>
<td>Defines the droplets used in a storm (raindrops, snowflakes, etc).</td>
</tr>
<tr>
<td><strong>ScatterSky</strong></td>
<td>Represents both the sun and sky for scenes with a dynamic time of day.</td>
</tr>
<tr>
<td><strong>SkyBox</strong></td>
<td>Represents the sky with an artist-created cubemap.</td>
</tr>
<tr>
<td><strong>Sun</strong></td>
<td>A global light affecting your entire scene and optionally renders a corona effect.</td>
</tr>
</tbody>
</table>
Detailed Description

Objects that represent the atmosphere and weather, such as the sky, sun, clouds, and precipitation.
Foliage
[Environment Objects]

Objects used for mass replication of foliage, such as grass, rocks, and bushes. More...
## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fxFoliageReplicator</td>
<td>An emitter to replicate fxFoliageItem objects across an area. <a href="#">More...</a></td>
</tr>
<tr>
<td>fxShapeReplicatedStatic</td>
<td>The object definition for shapes that will be replicated across an area using an fxShapeReplicator. <a href="#">More...</a></td>
</tr>
<tr>
<td>fxShapeReplicator</td>
<td>An emitter for objects to replicate across an area. <a href="#">More...</a></td>
</tr>
<tr>
<td>GroundCover</td>
<td>Covers the ground in a field of objects (IE: Grass, Flowers, etc). <a href="#">More...</a></td>
</tr>
</tbody>
</table>
## Functions

<table>
<thead>
<tr>
<th>void</th>
<th>StartClientReplication ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Activates the shape replicator.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>StartFoliageReplication ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Activates the foliage replicator.</td>
</tr>
</tbody>
</table>
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>$pref::GroundCover::densityScale</td>
<td>A global LOD scalar which can reduce the overall density of placed GroundCover.</td>
</tr>
<tr>
<td>static int</td>
<td>GroundCover::renderedBatches</td>
<td>Stat for number of rendered billboard batches.</td>
</tr>
<tr>
<td>static int</td>
<td>GroundCover::renderedBillboards</td>
<td>Stat for number of rendered billboards.</td>
</tr>
<tr>
<td>static int</td>
<td>GroundCover::renderedCells</td>
<td>Stat for number of rendered cells.</td>
</tr>
<tr>
<td>static int</td>
<td>GroundCover::renderedShapes</td>
<td>Stat for number of rendered shapes.</td>
</tr>
</tbody>
</table>
Detailed Description

Objects used for mass replication of foliage, such as grass, rocks, and bushes.
Function Documentation

void StartClientReplication( )

Activates the shape replicator.

Example:

    // Call the function
    StartClientReplication();

void StartFoliageReplication( )

Activates the foliage replicator.

Example:

    // Call the function
    StartFoliageReplication();
Variable Documentation

float $pref::GroundCover::densityScale

A global LOD scalar which can reduce the overall density of placed GroundCover.

int GroundCover::renderedBatches [static, inherited]

Stat for number of rendered billboard batches.

int GroundCover::renderedBillboards [static, inherited]

Stat for number of rendered billboards.

int GroundCover::renderedCells [static, inherited]

Stat for number of rendered cells.

int GroundCover::renderedShapes [static, inherited]

Stat for number of rendered shapes.
Forest
[Environment Objects]

Objects for efficiently placing and rendering trees, rocks, foliage, or any such feature needed in large number. More...
## Classes

<table>
<thead>
<tr>
<th>class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>class <strong>Forest</strong></td>
<td>Forest is a global-bounds scene object provides collision and rendering for a (.forest) data file. More...</td>
</tr>
<tr>
<td>class <strong>ForestBrushElement</strong></td>
<td>Represents a type of ForestItem and parameters for how it is placed when painting with a ForestBrush that contains it. More...</td>
</tr>
<tr>
<td>class <strong>ForestItemData</strong></td>
<td>Base class for defining a type of ForestItem. It does not implement loading or rendering of the shapeFile. More...</td>
</tr>
<tr>
<td>class <strong>ForestWindEmitter</strong></td>
<td>Object responsible for simulating wind in a level. More...</td>
</tr>
<tr>
<td>class <strong>TSForestItemData</strong></td>
<td>Concrete implementation of ForestItemData which loads and renders dts format shapeFiles. More...</td>
</tr>
</tbody>
</table>
**Variables**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td><strong>Forest::disableImposters</strong></td>
</tr>
<tr>
<td></td>
<td>A debugging aid which will disable rendering of all imposters in the forest.</td>
</tr>
<tr>
<td>static bool</td>
<td><strong>Forest::drawBounds</strong></td>
</tr>
<tr>
<td></td>
<td>A debugging aid which renders the forest bounds.</td>
</tr>
<tr>
<td>static bool</td>
<td><strong>Forest::drawCells</strong></td>
</tr>
<tr>
<td></td>
<td>A debugging aid which renders the forest cell bounds.</td>
</tr>
<tr>
<td>static bool</td>
<td><strong>Forest::forceImposters</strong></td>
</tr>
<tr>
<td></td>
<td>A debugging aid which will force all forest items to be rendered as imposters.</td>
</tr>
</tbody>
</table>
Detailed Description

Objects for efficiently placing and rendering trees, rocks, foliage, or any such feature needed in large number.
Variable Documentation

**bool Forest::disableImposters** [static, inherited]

A debugging aid which will disable rendering of all imposters in the forest.

**bool Forest::drawBounds** [static, inherited]

A debugging aid which renders the forest bounds.

**bool Forest::drawCells** [static, inherited]

A debugging aid which renders the forest cell bounds.

**bool Forest::forceImposters** [static, inherited]

A debugging aid which will force all forest items to be rendered as imposters.

Copyright © GarageGames, LLC. All Rights Reserved.
Miscellaneous

[Environment Objects]

Miscellaneous environmental and level objects. More...
## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ConvexShape</strong></td>
<td>A renderable, collidable convex shape defined by a collection of surface planes.</td>
</tr>
<tr>
<td><strong>LevelInfo</strong></td>
<td>Stores and controls the rendering and status information for a game level.</td>
</tr>
<tr>
<td><strong>Marker</strong></td>
<td>A single joint, or knot, along a path. Should be stored inside a <strong>Path</strong> container object. A path markers can be one of three primary movement types: &quot;normal&quot;, &quot;Position Only&quot;, or &quot;Kink&quot;.</td>
</tr>
<tr>
<td><strong>MissionArea</strong></td>
<td>Level object which defines the boundaries of the level.</td>
</tr>
<tr>
<td><strong>MissionMarker</strong></td>
<td>This is a base class for all &quot;marker&quot; related objects. It is a 3D representation of a point in the level.</td>
</tr>
<tr>
<td><strong>MissionMarkerData</strong></td>
<td>A very basic class containing information used by <strong>MissionMarker</strong> objects for rendering.</td>
</tr>
<tr>
<td><strong>OcclusionVolume</strong></td>
<td>An invisible shape that causes objects hidden from view behind it to not be rendered.</td>
</tr>
<tr>
<td><strong>Path</strong></td>
<td>A spline along which various objects can move along. The spline object acts like a container for <strong>Marker</strong> objects, which make up the joints, or knots, along the path. Paths can be assigned a speed, can be looping or non-looping. Each of a path’s markers can be one of three primary movement types: &quot;normal&quot;, &quot;Position Only&quot;, or &quot;Kink&quot;.</td>
</tr>
<tr>
<td><strong>PhysicalZone</strong></td>
<td></td>
</tr>
</tbody>
</table>
Physical Zones are areas that modify the player's gravity and/or velocity and/or applied force. [More...]

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Portal</td>
<td>An object that provides a &quot;window&quot; into a zone, allowing a viewer to see what's rendered in the zone. [More...]</td>
</tr>
<tr>
<td>Prefab</td>
<td>A collection of arbitrary objects which can be allocated and manipulated as a group. [More...]</td>
</tr>
<tr>
<td>ReflectorDesc</td>
<td>A datablock which defines performance and quality properties for dynamic reflections. [More...]</td>
</tr>
<tr>
<td>SpawnSphere</td>
<td>This class is used for creating any type of game object, assigning it a class, datablock, and other properties when it is spawned. [More...]</td>
</tr>
<tr>
<td>TerrainMaterial</td>
<td>The TerrainMaterial class organizes the material settings for a single terrain material layer. [More...]</td>
</tr>
<tr>
<td>TimeOfDay</td>
<td>Environmental object that triggers a day/night cycle in level. [More...]</td>
</tr>
<tr>
<td>WayPoint</td>
<td>Special type of marker, distinguished by a name and team ID number. [More...]</td>
</tr>
<tr>
<td>Zone</td>
<td>An object that represents an interior space. [More...]</td>
</tr>
</tbody>
</table>
Enumerations

```csharp
enum MarkerKnotType {
    Normal,
    Only,
    Kink
}

The type of knot that this marker will be.

More...
```

```csharp
enum MarkerSmoothingType {
    Spline,
    Linear
}

The type of smoothing this marker will have for pathed objects.

More...
```
Functions

**MissionArea  getMissionAreaServerObject ()**

Get the MissionArea object, if any.
Detailed Description

Miscellaneous environmental and level objects.
Enumeration Type Documentation

`enum MarkerKnotType`

The type of knot that this marker will be.

**Enumerator:**

- **Normal** Knot will have a smooth camera translation/rotation effect.
- **Only** Will do the same for translations, leaving rotation untouched.
- **Kink** The rotation will take effect immediately for an abrupt rotation change.

`enum MarkerSmoothingType`

The type of smoothing this marker will have for pathed objects.

**Enumerator:**

- **Spline** Marker will cause the movements of the pathed object to be smooth.
- **Linear** Marker will have no smoothing effect.
Function Documentation

MissionArea getMissionAreaServerObject()

Get the MissionArea object, if any.
Terrain
[Environment Objects]

Objects that specialize in representing terrain and other collidable/walkable surfaces. More...
### Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DecalRoad</td>
<td>A strip shaped decal defined by spine nodes which clips against Terrain objects. <a href="#">More...</a></td>
</tr>
<tr>
<td>GroundPlane</td>
<td>An infinite plane extending in all direction. <a href="#">More...</a></td>
</tr>
<tr>
<td>MeshRoad</td>
<td>A strip of rectangular mesh segments defined by a 3D spline for prototyping road-shaped objects in your scene. <a href="#">More...</a></td>
</tr>
<tr>
<td>TerrainBlock</td>
<td>Represent a terrain object in a Torque 3D level. <a href="#">More...</a></td>
</tr>
</tbody>
</table>
## Functions

<table>
<thead>
<tr>
<th>bool</th>
<th><code>getTerrainHeight (Point2I position)</code></th>
<th>Gets the terrain height at the specified position.</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>getTerrainHeight (F32 x, F32 y)</code></td>
<td>Gets the terrain height at the specified position.</td>
</tr>
<tr>
<td>bool</td>
<td><code>getTerrainHeightBelowPosition (Point2I position)</code></td>
<td>Takes a world point and find the &quot;highest&quot; terrain underneath it.</td>
</tr>
<tr>
<td>bool</td>
<td><code>getTerrainHeightBelowPosition (F32 x, F32 y)</code></td>
<td>Takes a world point and find the &quot;highest&quot; terrain underneath it.</td>
</tr>
<tr>
<td>bool</td>
<td><code>getTerrainUnderWorldPoint (Point3F position)</code></td>
<td>Gets the terrain block that is located under the given world point.</td>
</tr>
<tr>
<td>bool</td>
<td><code>getTerrainUnderWorldPoint (F32 x, F32 y, F32 z)</code></td>
<td>Takes a world point and find the &quot;highest&quot; terrain underneath it.</td>
</tr>
</tbody>
</table>
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| static bool | TerrainBlock::debugRender  
Triggers debug rendering of terrain cells. |
| float     | $pref::Terrain::detailScale  
A global detail scale used to tweak the material detail distances. |
| float     | $pref::Terrain::lodScale  
A global LOD scale used to tweak the default terrain screen error value. |
Detailed Description

Objects that specialize in representing terrain and other collidable/walkable surfaces.
**Function Documentation**

```markdown
bool getTerrainHeight(Point2I position)
```

Gets the terrain height at the specified position.

**Parameters:**

- `position` The world space point, minus the z (height) value. Formatted as ("x y")

**Returns:**

Returns the terrain height at the given point as an F32 value.

```markdown
bool getTerrainHeight(F32 x, F32 y)
```

Gets the terrain height at the specified position.

**Parameters:**

- `x` The X coordinate in world space
- `y` The Y coordinate in world space

**Returns:**

Returns the terrain height at the given point as an F32 value.

```markdown
bool getTerrainHeightBelowPosition(Point2I position)
```

Takes a world point and find the "highest" terrain underneath it.

**Parameters:**

- `position` The world space point, minus the z (height) value. Formatted as ("x y")
**Returns:**
Returns the closest terrain height below the given point as an F32 value.

```c
bool getTerrainHeightBelowPosition(F32 x, F32 y)
```

Takes a world point and find the "highest" terrain underneath it.

**Parameters:**
- `x` The X coordinate in world space
- `y` The Y coordinate in world space

**Returns:**
Returns the closest terrain height below the given point as an F32 value.

```c
bool getTerrainUnderWorldPoint(Point3F position)
```

Gets the terrain block that is located under the given world point.

**Parameters:**
- `position` The world space coordinate you wish to query at.
  Formatted as ("x y z")

**Returns:**
Returns the ID of the requested terrain block (0 if not found).

```c
bool getTerrainUnderWorldPoint(F32 x, F32 y, F32 z)
```
Takes a world point and find the "highest" terrain underneath it.

**Parameters:**
- $x$ The X coordinate in world space
- $y$ The Y coordinate in world space
- $z$ The Z coordinate in world space

**Returns:**
Returns the ID of the requested terrain block (0 if not found).
Variable Documentation

bool TerrainBlock::debugRender [static, inherited]

Triggers debug rendering of terrain cells.

float $pref::Terrain::detailScale

A global detail scale used to tweak the material detail distances.

float $pref::Terrain::lodScale

A global LOD scale used to tweak the default terrain screen error value.
Water
[Environment Objects]

Objects that represent water features, from puddles to rivers and oceans. More...
## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>River</strong></td>
<td>A water volume defined by a 3D spline. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>WaterBlock</strong></td>
<td>A block shaped water volume defined by a 3D scale and orientation. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>WaterObject</strong></td>
<td>Abstract base class for representing a body of water. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>WaterPlane</strong></td>
<td>Represents a large body of water stretching to the horizon in all directions. <a href="#">More...</a></td>
</tr>
<tr>
<td>Type</td>
<td>Variable</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>bool</td>
<td>$pref::Water::disableTrueReflections</td>
</tr>
<tr>
<td>static bool</td>
<td>WaterObject::wireframe</td>
</tr>
</tbody>
</table>
Detailed Description

Objects that represent water features, from puddles to rivers and oceans.
Variable Documentation

bool $pref::Water::disableTrueReflections

Force all water objects to use static cubemap reflections.

bool WaterObject::wireframe [static, inherited]

If true, will render the wireframe of the WaterObject.
Examples
## Classes

<table>
<thead>
<tr>
<th>class</th>
<th>RenderMeshExample</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>An example scene object which renders a mesh. <a href="#">More...</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th>RenderObjectExample</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>An example scene object which renders using a callback. <a href="#">More...</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th>RenderShapeExample</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>An example scene object which renders a DTS. <a href="#">More...</a></td>
</tr>
</tbody>
</table>
Detailed Description

Copyright © GarageGames, LLC. All Rights Reserved.
GUI

Subsystem to display user interface elements and handle high-level rendering control flow. More...
## Modules

<table>
<thead>
<tr>
<th><strong>3D Controls</strong></th>
<th>Controls to render 3D elements.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Core Controls</strong></td>
<td>Core parts of the Gui System.</td>
</tr>
<tr>
<td><strong>Button Controls</strong></td>
<td>A collection of various buttons (push buttons, radio buttons, check boxes, etc).</td>
</tr>
<tr>
<td><strong>General Controls</strong></td>
<td>A collection of general controls (bitmap, text, popup, etc).</td>
</tr>
<tr>
<td><strong>Container Controls</strong></td>
<td>A collection of various containers (container, window, scroll, etc).</td>
</tr>
<tr>
<td><strong>Image and Video Controls</strong></td>
<td>Controls that display images or videos.</td>
</tr>
<tr>
<td><strong>Value Controls</strong></td>
<td></td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Controls</td>
<td>Controls that display values and optionally allow to edit them.</td>
</tr>
<tr>
<td><strong>Utility Controls</strong></td>
<td>A collection of utility classes that support other GUI controls.</td>
</tr>
<tr>
<td><strong>Game Controls</strong></td>
<td>GUI controls dedicated to game play systems, such as heads up displays.</td>
</tr>
</tbody>
</table>
Detailed Description

Subsystem to display user interface elements and handle high-level rendering control flow.
Internationalization

Copyright © GarageGames, LLC. All Rights Reserved.
3D Controls
[GUI]

Controls to render 3D elements. More...
# Classes

<table>
<thead>
<tr>
<th>class</th>
<th>GameTSCtrl</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The main 3D viewport for a Torque 3D game.</td>
</tr>
<tr>
<td></td>
<td><a href="#">More...</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th>GuiObjectView</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GUI control which displays a 3D model.</td>
</tr>
<tr>
<td></td>
<td><a href="#">More...</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th>GuiTSCtrl</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Abstract base class for controls that render 3D scenes.</td>
</tr>
<tr>
<td></td>
<td><a href="#">More...</a></td>
</tr>
</tbody>
</table>
Detailed Description

Controls to render 3D elements.
Button Controls
[GUI]

A collection of various buttons (push buttons, radio buttons, check boxes, etc). More...
## Classes

<table>
<thead>
<tr>
<th>Class Name</th>
<th>Description</th>
<th>More...</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>GuiBitmapButtonCtrl</code></td>
<td>A button that renders its various states (mouse over, pushed, etc.) from separate bitmaps.</td>
<td></td>
</tr>
<tr>
<td><code>GuiBitmapButtonTextCtrl</code></td>
<td>An extension of <code>GuiBitmapButtonCtrl</code> that additionally renders a text label on the bitmapped button.</td>
<td></td>
</tr>
<tr>
<td><code>GuiBorderButtonCtrl</code></td>
<td>A push button that renders only a border.</td>
<td></td>
</tr>
<tr>
<td><code>GuiButtonBaseCtrl</code></td>
<td>The base class for the various button controls.</td>
<td></td>
</tr>
<tr>
<td><code>GuiButtonCtrl</code></td>
<td>The most widely used button class.</td>
<td></td>
</tr>
<tr>
<td><code>GuiCheckBoxCtrl</code></td>
<td>A named checkbox that can be toggled on and off.</td>
<td></td>
</tr>
<tr>
<td><code>GuiRadioCtrl</code></td>
<td>A button based around the radio concept.</td>
<td></td>
</tr>
<tr>
<td><code>GuiSwatchButtonCtrl</code></td>
<td>A button that is used to represent color; often used in correlation with a color picker.</td>
<td></td>
</tr>
<tr>
<td><code> GuiToggleButtonCtrl</code></td>
<td>Deprecated gui control.</td>
<td></td>
</tr>
</tbody>
</table>
Enumerations

```
enum GuiButtonType {
    PushButton,
    ToggleButton,
    RadioButton
}

Type of button control.

More...
```
**Detailed Description**

A collection of various buttons (push buttons, radio buttons, check boxes, etc).
Enumeration Type Documentation

enum GuiButtonType

Type of button control.

**Enumerator:**

*PushButton*  
A button that triggers an action when clicked.

*ToggleButton*  
A button that is toggled between on and off state.

*RadioButton*  
A button placed in groups for presenting choices.

Copyright © GarageGames, LLC. All Rights Reserved.
Container Controls
[GUI]

A collection of various containers (container, window, scroll, etc).
More...
## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
<th>More...</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GuiAutoScrollCtrl</strong></td>
<td>A container that scrolls its child control up over time.</td>
<td></td>
</tr>
<tr>
<td><strong>GuiContainer</strong></td>
<td>Brief Desc.</td>
<td></td>
</tr>
<tr>
<td><strong>GuiControlArrayControl</strong></td>
<td>A container that arranges children into a grid.</td>
<td></td>
</tr>
<tr>
<td><strong>GuiDynamicCtrlArrayControl</strong></td>
<td>A container that arranges children into a grid.</td>
<td></td>
</tr>
<tr>
<td><strong>GuiFrameSetCtrl</strong></td>
<td>A gui control allowing a window to be subdivided into panes, each of which displays a gui control child of the GuiFrameSetCtrl.</td>
<td></td>
</tr>
<tr>
<td><strong>GuiPaneControl</strong></td>
<td>A collapsible pane control.</td>
<td></td>
</tr>
<tr>
<td><strong>GuiPanel</strong></td>
<td>The GuiPanel panel is a container that when opaque will draw a left to right gradient using its profile fill and fill highlight colors.</td>
<td></td>
</tr>
<tr>
<td><strong>GuiRolloutCtrl</strong></td>
<td>A container that shows a single child with an optional header bar that can be used to collapse and expand the rollout.</td>
<td></td>
</tr>
<tr>
<td><strong>GuiScrollCtrl</strong></td>
<td>A container that allows to view one or more possibly larger controls inside its area by providing horizontal and/or vertical scroll bars.</td>
<td></td>
</tr>
<tr>
<td><strong>GuiSpeedometerHud</strong></td>
<td>Displays the speed of the current Vehicle based control object.</td>
<td></td>
</tr>
<tr>
<td><strong>GuiSplitContainer</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>GuiStackControl</td>
<td>A container that splits its area between two child controls. More...</td>
<td></td>
</tr>
<tr>
<td>GuiTabBookCtrl</td>
<td>A container that stacks its children horizontally or vertically. More...</td>
<td></td>
</tr>
<tr>
<td>GuiTabPageCtrl</td>
<td>A single page in a GuiTabBookCtrl. More...</td>
<td></td>
</tr>
<tr>
<td>GuiTreeViewCtrl</td>
<td>Hierarchical list of text items with optional icons. More...</td>
<td></td>
</tr>
<tr>
<td>GuiWindowCtrl</td>
<td>A window with a title bar and an optional set of buttons. More...</td>
<td></td>
</tr>
</tbody>
</table>
### Enumerations

<table>
<thead>
<tr>
<th>Enum</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GuiAutoScrollDirection</td>
<td>Direction in which to scroll the child control.</td>
</tr>
<tr>
<td>GuiDockingType</td>
<td>E.g. None, Client, Top, Bottom, Left, Right</td>
</tr>
<tr>
<td>GuiFrameState</td>
<td>alwaysOn, alwaysOff, dynamic</td>
</tr>
<tr>
<td>GuiHorizontalStackingType</td>
<td>Right, Left</td>
</tr>
<tr>
<td>GuiScrollBarBehavior</td>
<td>alwaysOn, alwaysOff</td>
</tr>
</tbody>
</table>

More...
**Dynamic**

Display behavior of a scroll bar. Determines when a scrollbar will be visible.

More...

```c
enum GuiSplitFixedPanel {
    None,
    FirstPanel,
    SecondPanel
}
```

Which side of the splitter to keep at a fixed size (if any).

More...

```c
enum GuiSplitOrientation {
    Vertical,
    Horizontal
}
```

Axis along which to divide the container's space.

More...

```c
enum GuiStackingType {
    Vertical,
    Horizontal,
    Dynamic
}
```

Stacking method used to position child controls.

More...

```c
enum GuiTabPosition {
    Top,
    Bottom
}
```
| enum       | GuiVerticalStackingType { |
|           |   Bottom,          |
|           |   Top             |
|           } |

Determines how child controls are stacked vertically.

More...
Detailed Description

A collection of various containers (container, window, scroll, etc).
### Enumeration Type Documentation

**enum GuiAutoScrollDirection**

Direction in which to scroll the child control.

**Enumerator:**

- **Up**  
  Scroll from bottom towards top.

- **Down**  
  Scroll from top towards bottom.

- **Left**  
  Scroll from right towards left.

- **Right**  
  Scroll from left towards right.

**enum GuiDockingType**

**Enumerator:**

- **None**
- **Client**
- **Top**
- **Bottom**
- **Left**
- **Right**

**enum GuiFrameState**

**Enumerator:**

- **alwaysOn**
alwaysOff
dynamic

**enum GuiHorizontalStackingType**

Determines how child controls are stacked horizontally.

**Enumerator:**

*Right*  Child controls are positioned in order from left to right (left-most control is first).

*Left*  Child controls are positioned in order from right to left (right-most control is first).

**enum GuiScrollBarBehavior**

Display behavior of a scroll bar. Determines when a scrollbar will be visible.

**Enumerator:**

*alwaysOn*  Always visible.

*alwaysOff*  Never visible.

*dynamic*  Only visible when actually needed, i.e. when the child control(s) exceed the visible space on the given axis.
enum GuiSplitFixedPanel

Which side of the splitter to keep at a fixed size (if any).

**Enumerator:**

- *None*  
  Allow both childs to resize (default).

- *FirstPanel*  
  Keep.

- *SecondPanel*

enum GuiSplitOrientation

Axis along which to divide the container's space.

**Enumerator:**

- *Vertical*  
  Divide vertically placing one child left and one child right.

- *Horizontal*  
  Divide horizontally placing one child on top and one child below.

enum GuiStackingType

Stacking method used to position child controls.

**Enumerator:**

- *Vertical*  
  Stack children vertically by setting their Y position.
Horizontal  Stack children horizontally by setting their X position.

Dynamic  Automatically switch between Vertical and Horizontal stacking. Vertical stacking is chosen when the stack control is taller than it is wide, horizontal stacking is chosen when the stack control is wider than it is tall.

---

**enum GuiTabPosition**

Where the control should put the tab headers for selecting individual pages.

**Enumerator:**

- **Top**  Tab headers on top edge.

- **Bottom**  Tab headers on bottom edge.

---

**enum GuiVerticalStackingType**

Determines how child controls are stacked vertically.

**Enumerator:**

- **Bottom**  Child controls are positioned in order from top to bottom (top-most control is first).

- **Top**  Child controls are positioned in order from bottom to top (bottom-most control is first).
Core Controls

[GUI]

Core parts of the Gui System. More...
classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GuiCanvas</td>
<td>A canvas on which rendering occurs.</td>
</tr>
<tr>
<td>GuiConsole</td>
<td>The on-screen, in-game console. Calls getLog() to get the on-screen console entries, then renders them as needed.</td>
</tr>
<tr>
<td>GuiConsoleEditCtrl</td>
<td>Text entry element of a GuiConsole.</td>
</tr>
<tr>
<td>GuiControl</td>
<td>Base class for all Gui control objects.</td>
</tr>
<tr>
<td>GuiControlProfile</td>
<td>A collection of properties that determine control behavior and rendering.</td>
</tr>
<tr>
<td>GuiCursor</td>
<td>Acts as a skin for the cursor, where each GuiCursor object can have its own look and click-zone.</td>
</tr>
<tr>
<td>GuiFadeinBitmapCtrl</td>
<td>A GUI control which renders a black square over a bitmap image. The black square will fade out, then fade back in after a determined time. This control is especially useful for transitions and splash screens.</td>
</tr>
<tr>
<td>GuiIconButtonCtrl</td>
<td>Draws the bitmap within a special button control. Only a single bitmap is used and the button will be drawn in a highlighted mode when the mouse hovers over it or when it has been clicked.</td>
</tr>
<tr>
<td>GuiListBoxCtrl</td>
<td>A list of text items.</td>
</tr>
<tr>
<td>GuiMenuBar</td>
<td>GUI Control which displays a horizontal bar with individual</td>
</tr>
</tbody>
</table>
drop-down menu items. Each menu item may also have submenu items. More...

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GuiMLTextCtrl</strong></td>
<td>A text control that uses the Gui Markup Language ('ML') tags to dynamically change the text. More...</td>
</tr>
<tr>
<td><strong>GuiMouseEventCtrl</strong></td>
<td>Used to overlaps a 'hot region' where you want to catch inputs with and have specific events occur based on individual callbacks. More...</td>
</tr>
<tr>
<td><strong>GuiTextCtrl</strong></td>
<td>GUI control object this displays a single line of text, without TorqueML. More...</td>
</tr>
<tr>
<td><strong>GuiTextEditSliderBitmapCtrl</strong></td>
<td>GUI Control which displays a numerical value which can be increased or decreased using a pair of bitmap up/down buttons. More...</td>
</tr>
<tr>
<td><strong>GuiTextEditSliderCtrl</strong></td>
<td>GUI Control which displays a numerical value which can be increased or decreased using a pair of arrows. More...</td>
</tr>
</tbody>
</table>
## Enumerations

<table>
<thead>
<tr>
<th>Enum</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>GuiAlignmentType</td>
<td>Left, Center, Right, Top, Bottom</td>
</tr>
<tr>
<td>GuiFontCharset</td>
<td>ANSI, SYMBOL, SHIFTJIS, HANGEUL, HANGUL, GB2312, CHINESEBIG5, OEM, JOHAB, HEBREW, ARABIC, GREEK, TURKISH, VIETNAMESE, THAI, EASTEUROPE, RUSSIAN, MAC, BALTIC</td>
</tr>
<tr>
<td>GuiHorizontalSizing</td>
<td>right, width, left, center, relative, windowRelative</td>
</tr>
</tbody>
</table>
Horizontal sizing behavior of a `GuiControl`.

More...

`enum GuiVerticalSizing {
    bottom,
    height,
    top,
    center,
    relative,
    windowRelative
}

Vertical sizing behavior of a `GuiControl`.

More...
### Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>excludeOtherInstance</code> (string <code>appIdentifier</code>)</td>
<td>Used to exclude/prevent all other instances using the same identifier specified.</td>
</tr>
<tr>
<td>string</td>
<td><code>StripMLControlChars</code> (string <code>inString</code>)</td>
<td>Strip TorqueML control characters from the specified string, returning a 'clean' version.</td>
</tr>
</tbody>
</table>
Variables

<table>
<thead>
<tr>
<th>GuiControl</th>
<th>$ThisControl</th>
</tr>
</thead>
<tbody>
<tr>
<td>The control for which a command is currently being evaluated. Only set during 'command' and altCommand callbacks to the control for which the command or altCommand is invoked.</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

Core parts of the Gui System.
Enumeration Type Documentation

enum GuiAlignmentType

Enumerator:

Left
Center
Right
Top
Bottom

enum GuiFontCharset

Enumerator:

ANSI
SYMBOL
SHIFTJIS
HANGEUL
HANGUL
GB2312
CHINESEBIG5
OEM
JOHAB
HEBREW
ARABIC
GREEK
TURKISH
TURKISH
VIETNAMESE
THAI
EASTEUROPE
RUSSIAN
MAC
BALTIC
enum GuiHorizontalSizing

Horizontal sizing behavior of a GuiControl.

**Enumerator:**

- right
- width
- left
- center
- relative
- windowRelative

enum GuiVerticalSizing

Vertical sizing behavior of a GuiControl.

**Enumerator:**

- bottom
- height
- top
- center
- relative
- windowRelative
**Function Documentation**

```c
bool excludeOtherInstance(string appIdentifier)
```

Used to exclude/prevent all other instances using the same identifier specified.

**Note:**

Not used on OSX, Xbox, or in Win debug builds

**Parameters:**

- `appIdentifier` Name of the app set up for exclusive use.

**Returns:**

False if another app is running that specified the same `appIdentifier`

---

```c
string StripMLControlChars(string inString)
```

Strip TorqueML control characters from the specified string, returning a 'clean' version.

**Parameters:**

- `inString` String to strip TorqueML control characters from.

**Example:**

```c
// Define the string to strip TorqueML control characters from
%string = "<font:Arial:24>How Now <color:c43c12>Brown <color:000000>Cow"

// Request the stripped version of the string
%strippedString = StripMLControlChars(%string)
```

**Returns:**
Version of the inputted string with all TorqueML characters removed.

See also:
References
Variable Documentation

**GuiControl $ThisControl**

The control for which a command is currently being evaluated. Only set during 'command' and altCommand callbacks to the control for which the command or altCommand is invoked.
Game Controls

[GUI]

GUI controls dedicated to game play systems, such as heads up displays. More...
### Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>GuiClockHud</code></td>
<td>Basic HUD clock. Displays the current simulation time offset from some base. <a href="#">More...</a></td>
</tr>
<tr>
<td><code>GuiCrossHairHud</code></td>
<td>Basic cross hair hud. Reacts to state of control object. Also displays health bar for named objects under the cross hair. <a href="#">More...</a></td>
</tr>
<tr>
<td><code>GuiGameListMenuCtrl</code></td>
<td>A base class for cross platform menu controls that are gamepad friendly. <a href="#">More...</a></td>
</tr>
<tr>
<td><code>GuiGameListMenuProfile</code></td>
<td>A <code>GuiControlProfile</code> with additional fields specific to <code>GuiGameListMenuCtrl</code>. <a href="#">More...</a></td>
</tr>
<tr>
<td><code>GuiGameListOptionsCtrl</code></td>
<td>A control for showing pages of options that are gamepad friendly. <a href="#">More...</a></td>
</tr>
<tr>
<td><code>GuiGameListOptionsProfile</code></td>
<td>A <code>GuiControlProfile</code> with additional fields specific to <code>GuiGameListOptionsCtrl</code>. <a href="#">More...</a></td>
</tr>
<tr>
<td><code>GuiHealthBarHud</code></td>
<td>A basic health bar. Shows the damage value of the current PlayerObjectType control object. <a href="#">More...</a></td>
</tr>
<tr>
<td><code>GuiHealthTextHud</code></td>
<td>Shows the health or energy value of the current PlayerObjectType control object. <a href="#">More...</a></td>
</tr>
<tr>
<td><code>GuiShapeNameHud</code></td>
<td>Displays name and damage of <code>ShapeBase</code> objects in its bounds. Must be a child of a <code>GuiTSCtrl</code> and a server connection must be present. <a href="#">More...</a></td>
</tr>
</tbody>
</table>
### Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void snapToggle ()</td>
<td>Prevents mouse movement from being processed.</td>
</tr>
</tbody>
</table>
Detailed Description

GUI controls dedicated to game play systems, such as heads up displays.
Function Documentation

void snapToggle( )

Prevents mouse movement from being processed.

In the source, whenever a mouse move event occurs GameTSCtrl::onMouseMove() is called. Whenever snapToggle() is called, it will flag a variable that can prevent this from happening: gSnapLine. This variable is not exposed to script, so you need to call this function to trigger it.

Example:

```cpp
// Snapping is off by default, so we will
// it on first:
PlayGui.snapToggle();

// Mouse movement should be disabled
// Let's turn it back on
PlayGui.snapToggle();
```
General Controls
[GUI]

A collection of general controls (bitmap, text, popup, etc). More...
### Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GuiBitmapCtrl</td>
<td>A gui control that is used to display an image.</td>
</tr>
<tr>
<td>GuiBubbleTextCtrl</td>
<td>A single-line text control that displays its text in a multi-line popup when clicked.</td>
</tr>
<tr>
<td>GuiDirectoryFileListCtrl</td>
<td>A control that displays a list of files from within a single directory in the game file system.</td>
</tr>
<tr>
<td>GuiMLTextEditCtrl</td>
<td>A text entry control that accepts the Gui Markup Language (‘ML’) tags and multiple lines.</td>
</tr>
<tr>
<td>GuiPopUpMenuCtrl</td>
<td>A control that allows to select a value from a drop-down list.</td>
</tr>
<tr>
<td>GuiPopUpMenuCtrlEx</td>
<td>A control that allows to select a value from a drop-down list.</td>
</tr>
<tr>
<td>GuiSeparatorCtrl</td>
<td>A control that renders a horizontal or vertical separator with an optional text label (horizontal only).</td>
</tr>
<tr>
<td>GuiTextEditCtrl</td>
<td>A component that places a text entry box on the screen.</td>
</tr>
<tr>
<td>GuiTextListCtrl</td>
<td>GUI control that displays a list of text. Text items in the list can be individually selected.</td>
</tr>
</tbody>
</table>
Enumerations

```cpp
enum GuiSeparatorType {
    Vertical,
    Horizontal
};

GuiSeparatorCtrl orientations.

More...
```
Detailed Description

A collection of general controls (bitmap, text, popup, etc).
### Enumeration Type Documentation

```cpp
enum GuiSeparatorType
```

**GuiSeparatorCtrl** orientations.

**Enumerator:**

- *Vertical*
- *Horizontal*

---

Copyright © GarageGames, LLC. All Rights Reserved.
Image and Video Controls

[GUI]

Controls that display images or videos. More...
# Classes

<table>
<thead>
<tr>
<th>class</th>
<th>GuiBitmapBorderCtrl</th>
<th>A control that renders a skinned border specified in its profile. More...</th>
</tr>
</thead>
<tbody>
<tr>
<td>class</td>
<td>GuiChunkedBitmapCtrl</td>
<td>This is a control that will render a specified bitmap or a bitmap specified in a referenced variable. More...</td>
</tr>
<tr>
<td>class</td>
<td>GuiTheoraCtrl</td>
<td>A control to playing Theora videos. More...</td>
</tr>
</tbody>
</table>
### Enumerations

#### GuiBitmapMode
```c
enum GuiBitmapMode {
   Stretched,
   Centered
}
```

Rendering behavior when placing bitmaps in controls.

#### GuiIconButtonIconLocation
```c
enum GuiIconButtonIconLocation {
   None,
   Left,
   Right,
   Center
}
```

#### GuiIconButtonTextLocation
```c
enum GuiIconButtonTextLocation {
   None,
   Bottom,
   Right,
   Top,
   Left,
   Center
}
```

#### GuiTheoraTranscoder
```c
enum GuiTheoraTranscoder {
   Auto,
   Generic,
   SSE2420RGBA
}
```

Routine to use for converting Theora's Y'CbCr pixel format to RGB color space.

---

More...
Detailed Description

Controls that display images or videos.
## Enumeration Type Documentation

### Enum GuiBitmapMode

Rendering behavior when placing Bitmaps in controls.

**Enumerator:**

- **Stretched**  Stretch Bitmap to fit control extents.
- **Centered**  Center Bitmap in control.

### Enum GuiIconButtonIconLocation

**Enumerator:**

- **None**
- **Left**
- **Right**
- **Center**

### Enum GuiIconButtonTextLocation

**Enumerator:**

- **None**
- **Bottom**
- **Right**
- **Top**
- **Left**
- **Center**
enum GuiTheoraTranscoder

Routine to use for converting Theora's Y'CbCr pixel format to RGB color space.

**Enumerator:**

*Auto*  
Automatically detect most appropriate setting.

*Generic*  
Slower but generic transcoder that can convert all Y'CbCr input formats to RGB or RGBA output.

*SSE2420RGBA*  
Fast SSE2-based transcoder with fixed conversion from 4:2:0 Y'CbCr to RGBA.
Utility Controls
[GUI]

A collection of utility classes that support other GUI controls. More...
<table>
<thead>
<tr>
<th>class</th>
<th>GuiDragAndDropControl</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A container control that can be used to implement drag&amp;drop behavior. <a href="#">More...</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th>GuiInputCtrl</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A control that locks the mouse and reports all keyboard input events to script. <a href="#">More...</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th>GuiMessageVectorCtrl</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A chat HUD control that displays messages from a MessageVector. <a href="#">More...</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th>GuiScriptNotifyCtrl</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A control which adds several reactions to other GUIs via callbacks. <a href="#">More...</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th>GuiTickCtrl</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Brief Description. <a href="#">More...</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th>MessageVector</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Store a list of chat messages. <a href="#">More...</a></td>
</tr>
</tbody>
</table>
Detailed Description

A collection of utility classes that support other GUI controls.

Copyright © GarageGames, LLC. All Rights Reserved.
Value Controls

[GUI]

Controls that display values and optionally allow to edit them. More...
### Classes

<table>
<thead>
<tr>
<th>class</th>
<th>GUIBankCtrl</th>
<th>A control that plots one or more curves in a chart. <a href="#">More...</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>class</td>
<td>GuiProgressBitmapCtrl</td>
<td>A horizontal progress bar rendered from a repeating image. <a href="#">More...</a></td>
</tr>
<tr>
<td>class</td>
<td>GuiProgressCtrl</td>
<td>GUI Control which displays a horizontal bar which increases as the progress value of 0.0 - 1.0 increases. <a href="#">More...</a></td>
</tr>
<tr>
<td>class</td>
<td>GuiSliderCtrl</td>
<td>A control that displays a value between its minimal and maximal bounds using a slider placed on a vertical or horizontal axis. <a href="#">More...</a></td>
</tr>
</tbody>
</table>
### Enumerations

```c
enum GuiGraphType {
    Bar,
    Filled,
    Point,
    PolyLine
};
```

The charting style of a single plotting curve in a `GuiGraphCtrl`.

[More...]
Detailed Description

Controls that display values and optionally allow to edit them.
Enumeration Type Documentation

enum GuiGraphType

The charting style of a single plotting curve in a GuiGraphCtrl.

Enumerator:

- **Bar**: Plot the curve as a bar chart.
- **Filled**: Plot a filled poly graph that connects the data points on the curve.
- **Point**: Plot each data point on the curve as a single dot.
- **PolyLine**: Plot straight lines through the data points of the curve.

Copyright © GarageGames, LLC. All Rights Reserved.
Game

This section includes all objects, functions, and variables related to game play elements exposed by Torque 3D. More...
**Classes**

<table>
<thead>
<tr>
<th>class</th>
<th>SimObject</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Base class for almost all objects involved in the simulation.</td>
</tr>
</tbody>
</table>

*More...*
# Modules

<table>
<thead>
<tr>
<th><strong>Game Objects</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Objects which can be controlled or directly interact with a user, such as <strong>Player</strong>, <strong>Projectile</strong>, <strong>Item</strong>, etc.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Special Effects</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Classes responsible for special effect objects, such as <strong>Explosion</strong>, <strong>Debris</strong>, <strong>Particles</strong>, etc.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>AI</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Classes and functions related to artificial intelligence for Torque 3D.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Miscellaneous</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Miscellaneous/uncategorized game related objects.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Physics</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Objects and functions related to Torque 3D's physics layer.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Vehicles</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>This section is dedicated to vehicle game objects, such as the base <strong>Vehicle</strong> class, <strong>WheeledVehicle</strong>, <strong>FlyingVehicle</strong>, and so on.</td>
</tr>
</tbody>
</table>
### Functions

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>addBadWord</code> (string badWord)</td>
<td>Add a string to the bad word filter.</td>
</tr>
<tr>
<td>bool</td>
<td><code>containerBoxEmpty</code> (int mask, Point3F center, float xRadius, float yRadius=-1, float zRadius=-1, bool useClientContainer=false)</td>
<td>See if any objects of the given types are present in box of given extent.</td>
</tr>
<tr>
<td>string</td>
<td><code>containerFindFirst</code> (int mask, Point3F point, float x, float y, float z)</td>
<td>Find objects matching the bitmask type within a box centered at point, with extents x, y, z.</td>
</tr>
<tr>
<td>string</td>
<td><code>containerFindNext</code> ()</td>
<td>Get more results from a previous call to <code>containerFindFirst()</code>.</td>
</tr>
<tr>
<td>string</td>
<td><code>containerRayCast</code> (Point3F start, Point3F end, int mask, SceneObject pExempt=NULL, bool useClientContainer=false)</td>
<td>Cast a ray from start to end, checking for collision against items matching mask.</td>
</tr>
<tr>
<td>float</td>
<td><code>containerSearchCurrDist</code> (bool useClientContainer=false)</td>
<td>Get distance of the center of the current item from the center of the current initContainerRadiusSearch.</td>
</tr>
<tr>
<td>float</td>
<td><code>containerSearchCurrRadiusDist</code> (bool useClientContainer=false)</td>
<td>Get the distance of the closest point of the current item from the center of the current initContainerRadiusSearch.</td>
</tr>
</tbody>
</table>
| SceneObject | `containerSearchNext` (bool useClientContainer=false) | Get next item from a search started with `initContainerRadiusSearch()` or `initContainerTypeSearch()`.


<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bool containsBadWords (string text)</code></td>
<td>Checks to see if text is a bad word.</td>
</tr>
<tr>
<td><code>string filterString (string baseString=NULL, string replacementChars=NULL)</code></td>
<td>Replaces the characters in a string with designated text.</td>
</tr>
<tr>
<td><code>void initContainerRadiusSearch (Point3F pos, float radius, int mask, bool useClientContainer=false)</code></td>
<td>Start a search for items at the given position and within the given radius, filtering by mask.</td>
</tr>
<tr>
<td><code>void initContainerTypeSearch (int mask, bool useClientContainer=false)</code></td>
<td>Start a search for all items of the types specified by the bitset mask.</td>
</tr>
<tr>
<td><code>bool isPointInside (Point3F position)</code></td>
<td>Check to see if a point in world space is inside of an interior.</td>
</tr>
<tr>
<td><code>bool isPointInside (F32 x, F32 y, F32 z)</code></td>
<td></td>
</tr>
<tr>
<td><code>void resetFPSTracker ()</code></td>
<td>Reset FPS stats (fps::).</td>
</tr>
<tr>
<td><code>void sceneDumpZoneStates (bool updateFirst=true)</code></td>
<td>Dump the current zoning states of all zone spaces in the scene to the console.</td>
</tr>
<tr>
<td><code>SceneObject sceneGetZoneOwner (int zoneld=true)</code></td>
<td>Return the <code>SceneObject</code> that contains the given zone.</td>
</tr>
<tr>
<td><code>void setInteriorRenderMode (int mode)</code></td>
<td>Globally changes how InteriorInstances are rendered. Useful for debugging geometry and rendering artifacts.</td>
</tr>
<tr>
<td><code>bool spawnObject (class[, dataBlock, name, properties, script])</code></td>
<td></td>
</tr>
</tbody>
</table>
Global function used for spawning any type of object.
## Variables

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>$cameraFov</td>
<td>The camera's Field of View.</td>
</tr>
<tr>
<td>float</td>
<td>$mvBackwardAction</td>
<td>Backwards movement speed for the active player.</td>
</tr>
<tr>
<td>bool</td>
<td>$mvDeviceIsKeyboardMouse</td>
<td>Boolean state for if the system is using a keyboard and mouse or not.</td>
</tr>
<tr>
<td>float</td>
<td>$mvDownAction</td>
<td>Downwards movement speed for the active player.</td>
</tr>
<tr>
<td>float</td>
<td>$mvForwardAction</td>
<td>Forwards movement speed for the active player.</td>
</tr>
<tr>
<td>bool</td>
<td>$mvFreeLook</td>
<td>Boolean state for if freelook is active or not.</td>
</tr>
<tr>
<td>float</td>
<td>$mvLeftAction</td>
<td>Left movement speed for the active player.</td>
</tr>
<tr>
<td>float</td>
<td>$mvPitch</td>
<td>Current pitch value, typically applied through input devices, such as a mouse.</td>
</tr>
<tr>
<td>float</td>
<td>$mvPitchDownSpeed</td>
<td>Downwards pitch speed.</td>
</tr>
<tr>
<td>float</td>
<td>$mvPitchUpSpeed</td>
<td>Upwards pitch speed.</td>
</tr>
<tr>
<td>float</td>
<td>$mvRightAction</td>
<td>Right movement speed for the active player.</td>
</tr>
<tr>
<td>float</td>
<td>$mvRoll</td>
<td>Current roll value, typically applied through input devices, such as a mouse.</td>
</tr>
<tr>
<td>float</td>
<td>$mvRollLeftSpeed</td>
<td>Left roll speed.</td>
</tr>
<tr>
<td>Type</td>
<td>Variable</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>float</td>
<td>$mvRollRightSpeed</td>
<td>Right roll speed.</td>
</tr>
<tr>
<td>int</td>
<td>$mvTriggerCount0</td>
<td>Used to determine the trigger counts of buttons. Namely used for input actions such as jumping and weapons firing.</td>
</tr>
<tr>
<td>int</td>
<td>$mvTriggerCount1</td>
<td>Used to determine the trigger counts of buttons. Namely used for input actions such as jumping and weapons firing.</td>
</tr>
<tr>
<td>int</td>
<td>$mvTriggerCount2</td>
<td>Used to determine the trigger counts of buttons. Namely used for input actions such as jumping and weapons firing.</td>
</tr>
<tr>
<td>int</td>
<td>$mvTriggerCount3</td>
<td>Used to determine the trigger counts of buttons. Namely used for input actions such as jumping and weapons firing.</td>
</tr>
<tr>
<td>int</td>
<td>$mvTriggerCount4</td>
<td>Used to determine the trigger counts of buttons. Namely used for input actions such as jumping and weapons firing.</td>
</tr>
<tr>
<td>int</td>
<td>$mvTriggerCount5</td>
<td>Used to determine the trigger counts of buttons. Namely used for input actions such as jumping and weapons firing.</td>
</tr>
<tr>
<td>float</td>
<td>$mvUpAction</td>
<td>Upwards movement speed for the active player.</td>
</tr>
<tr>
<td>float</td>
<td>$mvXAxis_L</td>
<td>Left thumbstick X axis position on a dual-analog gamepad.</td>
</tr>
<tr>
<td>float</td>
<td>$mvXAxis_R</td>
<td>Right thumbstick X axis position on a dual-analog gamepad.</td>
</tr>
<tr>
<td>float</td>
<td>$mvYaw</td>
<td>Current yaw value, typically applied through input devices, such as a mouse.</td>
</tr>
<tr>
<td>float</td>
<td>$mvYawLeftSpeed</td>
<td>Left Yaw speed.</td>
</tr>
<tr>
<td>float</td>
<td>$mvYawRightSpeed</td>
<td></td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>$mvYAxis_L</td>
<td>Left thumbstick Y axis position on a dual-analog gamepad.</td>
<td></td>
</tr>
<tr>
<td>$mvYAxis_R</td>
<td>Right thumbstick Y axis position on a dual-analog gamepad.</td>
<td></td>
</tr>
<tr>
<td>$Ease::Back</td>
<td>Backwards ease for curve movement.</td>
<td></td>
</tr>
<tr>
<td>$Ease::Bounce</td>
<td>Bounce ease for curve movement.</td>
<td></td>
</tr>
<tr>
<td>$Ease::Circular</td>
<td>Circular ease for curve movement.</td>
<td></td>
</tr>
<tr>
<td>$Ease::Cubic</td>
<td>Cubic ease for curve movement.</td>
<td></td>
</tr>
<tr>
<td>$pref::Camera::distanceScale</td>
<td>A scale to apply to the normal visible distance, typically used for tuning performance.</td>
<td></td>
</tr>
<tr>
<td>$Ease::Elastic</td>
<td>Elastic ease for curve movement.</td>
<td></td>
</tr>
<tr>
<td>$pref::enableBadWordFilter</td>
<td>If true, the bad word filter will be enabled.</td>
<td></td>
</tr>
<tr>
<td>$pref::enablePostEffects</td>
<td>If true, post effects will be enabled.</td>
<td></td>
</tr>
<tr>
<td>$Ease::Exponential</td>
<td>Exponential ease for curve movement.</td>
<td></td>
</tr>
<tr>
<td>$Ease::In</td>
<td>In ease for curve movement.</td>
<td></td>
</tr>
<tr>
<td>$Ease::InOut</td>
<td>InOut ease for curve movement.</td>
<td></td>
</tr>
<tr>
<td>$pref::Input::JoystickEnabled</td>
<td>If true, the joystick is currently enabled.</td>
<td></td>
</tr>
<tr>
<td>int</td>
<td>$Ease::Linear</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Linear ease for curve movement.</td>
<td></td>
</tr>
<tr>
<td>int</td>
<td>$Ease::Out</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Out ease for curve movement.</td>
<td></td>
</tr>
<tr>
<td>int</td>
<td>$Ease::Quadratic</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Quadratic ease for curve movement.</td>
<td></td>
</tr>
<tr>
<td>int</td>
<td>$Ease::Quartic</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Quartic ease for curve movement.</td>
<td></td>
</tr>
<tr>
<td>int</td>
<td>$Ease::Quintic</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Quintic ease for curve movement.</td>
<td></td>
</tr>
<tr>
<td>int</td>
<td>$Ease::Sinusoidal</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Sinusoidal ease for curve movement.</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

This section includes all objects, functions, and variables related to game play elements exposed by Torque 3D.
**Function Documentation**

```csharp
bool addBadWord(string badWord )
```

Add a string to the bad word filter.

The bad word filter is a table containing words which will not be displayed in chat windows. Instead, a designated replacement string will be displayed. There are already a number of bad words automatically defined.

**Parameters:**

- `badWord` Exact text of the word to restrict.

**Returns:**

- True if word was successfully added, false if the word or a subset of it already exists in the table

**See also:**

- `filterString()`

**Example:**

```csharp
// In this game, "Foobar" is banned
%badWord = "Foobar";

// Returns true, word was successfully added
addBadWord(%badWord);

// Returns false, word has already been added
addBadWord("Foobar");
```

```csharp
bool containerBoxEmpty(int mask,
                       Point3F center,
                       float xRadius,
                       ...)
```
float yRadius = -1,
float zRadius = -1,
bool useClientContainer = false)

See if any objects of the given types are present in box of given extent.

**Note:**
Extent parameter is last since only one radius is often needed. If one radius is provided, the yRadius and zRadius are assumed to be the same. Unfortunately, if you need to use the client container, you'll need to set all of the radius parameters. Fortunately, this function is mostly used on the server.

**Parameters:**
- *mask* indicates the type of objects we are checking against.
- *center* center of box.
- *xRadius* search radius in the x-axis. See note above.
- *yRadius* search radius in the y-axis. See note above.
- *zRadius* search radius in the z-axis. See note above.
- *useClientContainer* optionally indicates the search should be within the client container.

**Returns:**
true if the box is empty, false if any object is found.

```cpp
string containerFindFirst(int mask, Point3F point, float x, float y,
                        float z, bool useClientContainer = false)
```
Find objects matching the bitmask type within a box centered at point, with extents x, y, z.

**Returns:**
- The first object found, or an empty string if nothing was found. Thereafter, you can get more results using `containerFindNext()`.

**See also:**
- `containerFindNext`

```c
string containerFindNext( )
```

Get more results from a previous call to `containerFindFirst()`.

**Note:**
- You must call `containerFindFirst()` to begin the search.

**Returns:**
- The next object found, or an empty string if nothing else was found.

**See also:**
- `containerFindFirst()`

```c
string containerRayCast(Point3F start, Point3F end, int mask, SceneObject pExempt = NULL, bool useClientContainer = false )
```
Cast a ray from start to end, checking for collision against items matching mask.

If pExempt is specified, then it is temporarily excluded from collision checks (For instance, you might want to exclude the player if said player was firing a weapon.)

**Parameters:**

- **start**
  - An XYZ vector containing the tail position of the ray.
- **end**
  - An XYZ vector containing the head position of the ray
- **mask**
  - A bitmask corresponding to the type of objects to check for
- **pExempt**
  - An optional ID for a single object that ignored for this raycast
- **useClientContainer**
  - Optionally indicates the search should be within the client container.

**Returns:**

A string containing either null, if nothing was struck, or these fields:
- The ID of the object that was struck.
- The x, y, z position that it was struck.
- The x, y, z of the normal of the face that was struck.
- The distance between the start point and the position we hit.

```c
float containerSearchCurrDist(bool useClientContainer = false)
```

Get distance of the center of the current item from the center of the current initContainerRadiusSearch.

**Parameters:**

- **useClientContainer**
  - Optionally indicates the search should be within the client container.
Returns:

distance from the center of the current object to the center of the search

See also:

containerSearchNext

---

float containerSearchCurrRadiusDist (bool useClientContainer = false)

Get the distance of the closest point of the current item from the center of the current initContainerRadiusSearch.

Parameters:

useClientContainer Optionally indicates the search should be within the client container.

Returns:

distance from the closest point of the current object to the center of the search

See also:

containerSearchNext

---

SceneObject containerSearchNext (bool useClientContainer = false)

Get next item from a search started with initContainerRadiusSearch() or initContainerTypeSearch().

Parameters:

useClientContainer Optionally indicates the search should be within the client container.

Returns:

the next object found in the search, or null if no more
Example:

```
// print the names of all nearby ShapeBase objects
%position = %obj.getPosition;
%radius = 20;
%mask = $TypeMasks::ShapeBaseObjectType;
initContainerRadiusSearch( %position, %radius, %mask);
while (%targetObject = containerSearchNext)
{
   echo( "Found: " @ %targetObject.getName() );
}
```

See also:

```
initContainerRadiusSearch()
initContainerTypeSearch()
```

```
bool containsBadWords(string text )
```

Checks to see if text is a bad word.

The text is considered to be a bad word if it has been added to the bad word filter.

**Parameters:**

```
text  Text to scan for bad words
```

**Returns:**

```
True if the text has bad word(s), false if it is clean
```

See also:

```
addBadWord()
filterString()
```

**Example:**

```
// In this game, "Foobar" is banned
```
%badWord = "Foobar";

// Add a banned word to the bad word filter
addBadWord(%badWord);

// Create the base string, can come from a
%userText = "Foobar";

// Create a string of random letters
%replacementChars = "knqwrtlzs";

// If the text contains a bad word, filter
// Otherwise print the original text
if(containsBadWords(%userText))
{
    // Filter the string
    %filteredText = filterString(%userText,

    // Print filtered text
    echo(%filteredText);
}
else
    echo(%userText);

string filterString(string baseString = NULL,
    string replacementChars = NULL)

Replaces the characters in a string with designated text.

Uses the bad word filter to determine which characters within the string will be replaced.
Parameters:

- **baseString**
  The original string to filter.
- **replacementChars**
  A string containing letters you wish to swap in the baseString.

Returns:

The new scrambled string

See also:

- addBadWord()
- containsBadWords()

Example:

```c
// Create the base string, can come from anywhere
%baseString = "Foobar";

// Create a string of random letters
%replacementChars = "knqwrths";

// Filter the string
%newString = filterString(%baseString, %replacementChars);

// Print the new string to console
echo(%newString);
```

```c
void initContainerRadiusSearch(Point3F pos,
                               float radius,
                               int mask,
                               bool useClientContainer = false)
```

Start a search for items at the given position and within the given radius, filtering by mask.
**Parameters:**

- `pos` Center position for the search
- `radius` Search radius
- `mask` Bitmask of object types to include in the search
- `useClientContainer` Optionally indicates the search should be within the client container.

**See also:**

- `containerSearchNext`

```cpp
void initContainerTypeSearch(int mask,
                              bool useClientContainer = false)
```

Start a search for all items of the types specified by the bitset mask.

**Parameters:**

- `mask` Bitmask of object types to include in the search
- `useClientContainer` Optionally indicates the search should be within the client container.

**See also:**

- `containerSearchNext`

```cpp
bool isPointInside(Point3F position)
```

Check to see if a point in world space is inside of an interior.

**Parameters:**

- `position` The position to check in world space.
Example:

```c++
// Check to see if a point is inside any :
%point = "100 100 100";
%isInside = isPointInside(%point);
```

```c++
bool isPointInside(F32 x,
                 F32 y,
                 F32 z
               )
```

Check to see if a set of coordinates in world space are inside of an interior.

**Parameters:**

- `x` X-coordinate for position in world space.
- `y` Y-coordinate for position in world space.
- `z` Z-coordinate for position in world space.

**Example:**

```c++
// Check to see if a point is inside any :
%isInside = isPointInside(100, 100, 100);
```

```c++
void resetFPSTracker( )
```

Reset FPS stats (fps::).

```c++
void sceneDumpZoneStates(bool updateFirst = true )
```

Dump the current zoning states of all zone spaces in the scene to the console.

**Parameters:**
If `true`, zoning states are brought up to date first; if `false`, the zoning states are dumped as is.

**Note:**
Only valid on the client.

```csharp
SceneObject sceneGetZoneOwner(int zoneld = true)
```

Return the `SceneObject` that contains the given zone.

**Parameters:**
- `zoneld` ID of zone.

**Returns:**
- A `SceneObject` or `NULL` if the given `zoneld` is invalid.

**Note:**
Only valid on the client.

```csharp
void setInteriorRenderMode(int mode)
```

Globally changes how InteriorInstances are rendered. Useful for debugging geometry and rendering artifacts.

**Note:**
This does not work in shipping mode

**Parameters:**
- `mode` The render mode can be one of the following numbers:
  - `NormalRender = 0`,
  - `NormalRenderLines = 1`,
  - `ShowDetail = 2`,
  - `ShowUsers_WithFacets = 3`
ShowAmbiguous = 3,
ShowOrphan = 4,
ShowLightmaps = 5,
ShowTexturesOnly = 6,
ShowPortalZones = 7,
ShowOutsideVisible = 8,
ShowCollisionFans = 9,
ShowStrips = 10,
ShowNullSurfaces = 11,
ShowLargeTextures = 12,
ShowHullSurfaces = 13,
ShowVehicleHullSurfaces = 14,
ShowVertexColors = 15,
ShowDetailLevel = 16

bool spawnObject(class [, dataBlock, name, properties, script] )

Global function used for spawning any type of object.

Note: This is separate from SpawnSphere::spawnObject(). This function is not called off any other class and uses different parameters than the SpawnSphere’s function. In the source, SpawnSphere::spawnObject() actually calls this function and passes its properties (spawnClass, spawnDatablock, etc).

Parameters:
class	Mandatory field specifying the object class, such as Player or TSStatic.
Field specifying the object's datablock, optional for objects such as TSStatic, mandatory for game objects like Player.

name	Optional field specifying a name for this instance of the object.

properties	Optional set of parameters applied to the spawn object during creation.

script	Optional command(s) to execute when spawning an object.

Example:

```plaintext
// Set the parameters for the spawn function
%objectClass = "Player";
%objectDatablock = "DefaultPlayerData";
%objectName = "PlayerName";
%additionalProperties = "health = "0";"
%spawnScript = "echo("Player Spawned")"
// Spawn with the engine's Sim::spawnObject
%player = spawnObject(%objectClass, %objectDatablock, %objectName, %additionalProperties, %spawnScript);
```
<table>
<thead>
<tr>
<th>Variable</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>float $cameraFov</td>
<td>The camera's Field of View.</td>
</tr>
<tr>
<td>float $mvBackwardAction</td>
<td>Backwards movement speed for the active player.</td>
</tr>
<tr>
<td>bool $mvDeviceIsKeyboardMouse</td>
<td>Boolean state for if the system is using a keyboard and mouse or not.</td>
</tr>
<tr>
<td>float $mvDownAction</td>
<td>Downwards movement speed for the active player.</td>
</tr>
<tr>
<td>float $mvForwardAction</td>
<td>Forwards movement speed for the active player.</td>
</tr>
<tr>
<td>bool $mvFreeLook</td>
<td>Boolean state for if freelook is active or not.</td>
</tr>
<tr>
<td>float $mvLeftAction</td>
<td></td>
</tr>
</tbody>
</table>
Left movement speed for the active player.

float $mvPitch

Current pitch value, typically applied through input devices, such as a mouse.

float $mvPitchDownSpeed

Downwards pitch speed.

float $mvPitchUpSpeed

Upwards pitch speed.

float $mvRightAction

Right movement speed for the active player.

float $mvRoll

Current roll value, typically applied through input devices, such as a mouse.

float $mvRollLeftSpeed

Left roll speed.
float $mvRollRightSpeed

Right roll speed.

int $mvTriggerCount0

Used to determine the trigger counts of buttons. Namely used for input actions such as jumping and weapons firing.

int $mvTriggerCount1

Used to determine the trigger counts of buttons. Namely used for input actions such as jumping and weapons firing.

int $mvTriggerCount2

Used to determine the trigger counts of buttons. Namely used for input actions such as jumping and weapons firing.

int $mvTriggerCount3

Used to determine the trigger counts of buttons. Namely used for input actions such as jumping and weapons firing.

int $mvTriggerCount4

Used to determine the trigger counts of buttons. Namely used for input actions such as jumping and weapons firing.

int $mvTriggerCount5
Used to determine the trigger counts of buttons. Namely used for input actions such as jumping and weapons firing.

**float $mvUpAction**

Upwards movement speed for the active player.

**float $mvXAxis_L**

Left thumbstick X axis position on a dual-analog gamepad.

**float $mvXAxis_R**

Right thumbstick X axis position on a dual-analog gamepad.

**float $mvYaw**

Current yaw value, typically applied through input devices, such as a mouse.

**float $mvYawLeftSpeed**

Left Yaw speed.

**float $mvYawRightSpeed**

Right Yaw speed.

**float $mvYAxis_L**
Left thumbstick Y axis position on a dual-analog gamepad.

float $mvYAxis_R

Right thumbstick Y axis position on a dual-analog gamepad.

int $Ease::Back

Backwards ease for curve movement.

int $Ease::Bounce

Bounce ease for curve movement.

int $Ease::Circular

Circular ease for curve movement.

int $Ease::Cubic

Cubic ease for curve movement.

float $pref::Camera::distanceScale

A scale to apply to the normal visible distance, typically used for tuning performance.

int $Ease::Elastic
Elastic ease for curve movement.

bool $pref::enableBadWordFilter

If true, the bad word filter will be enabled.

bool $pref::enablePostEffects

If true, post effects will be enabled.

int $Ease::Exponential

Exponential ease for curve movement.

int $Ease::In

In ease for curve movement.

int $Ease::InOut

InOut ease for curve movement.

bool $pref::Input::JoystickEnabled

If true, the joystick is currently enabled.

int $Ease::Linear

Linear ease for curve movement.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int $Ease::Out</td>
<td>Out ease for curve movement.</td>
</tr>
<tr>
<td>int $Ease::Quadratic</td>
<td>Quadratic ease for curve movement.</td>
</tr>
<tr>
<td>int $Ease::Quartic</td>
<td>Quartic ease for curve movement.</td>
</tr>
<tr>
<td>int $Ease::Quintic</td>
<td>Quintic ease for curve movement.</td>
</tr>
<tr>
<td>int $Ease::Sinusoidal</td>
<td>Sinusoidal ease for curve movement.</td>
</tr>
</tbody>
</table>
AI
[Game]

Classes and functions related to artificial intelligence for Torque 3D.
More...
### Classes

<table>
<thead>
<tr>
<th>class</th>
<th>AIConnection</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Simulated client driven by AI commands.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th>AIConnection</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Special client connection driven by an AI, rather than a human.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th>AIPlayer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A Player object not controlled by conventional input, but by an AI engine.</td>
</tr>
</tbody>
</table>
Functions

int aiConnect (...)  
Creates a new AlConnection, and passes arguments to its onConnect script callback.
Detailed Description

Classes and functions related to artificial intelligence for Torque 3D.
Function Documentation

int aiConnect( ... )

Creates a new AIConnection, and passes arguments to its onConnect script callback.

Returns:
   The newly created AIConnection

See also:
   GameConnection for parameter information
Game Objects

Objects which can be controlled or directly interact with a user, such as Player, Projectile, Item, etc. More...
## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
<th>More...</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AIPlayer</strong></td>
<td>A Player object not controlled by conventional input, but by an AI engine.</td>
<td></td>
</tr>
<tr>
<td><strong>AITurretShape</strong></td>
<td>Provides an AI controlled turret.</td>
<td></td>
</tr>
<tr>
<td><strong>AITurretShapeData</strong></td>
<td>Defines properties for an AITurretShape object.</td>
<td></td>
</tr>
<tr>
<td><strong>GameBase</strong></td>
<td>Base class for game objects which use datablocks, networking, are editable, and need to process ticks.</td>
<td></td>
</tr>
<tr>
<td><strong>GameBaseData</strong></td>
<td>Scriptable, demo-able datablock. Used by GameBase objects.</td>
<td></td>
</tr>
<tr>
<td><strong>InteriorInstance</strong></td>
<td>Object used to represent buildings and other architectural structures (legacy).</td>
<td></td>
</tr>
<tr>
<td><strong>Item</strong></td>
<td>Base Item class. Uses the ItemData datablock for common properties.</td>
<td></td>
</tr>
<tr>
<td><strong>ItemData</strong></td>
<td>Stores properties for an individual Item type.</td>
<td></td>
</tr>
<tr>
<td><strong>Player</strong></td>
<td>A client-controlled player character.</td>
<td></td>
</tr>
<tr>
<td><strong>PlayerData</strong></td>
<td>Defines properties for a Player object.</td>
<td></td>
</tr>
<tr>
<td><strong>Projectile</strong></td>
<td>Base projectile class. Uses the ProjectileData class for properties of individual projectiles.</td>
<td></td>
</tr>
<tr>
<td><strong>ProjectileData</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>ProximityMine</td>
<td>Stores properties for an individual projectile type.</td>
<td></td>
</tr>
<tr>
<td>SceneObject</td>
<td>A networkable object that exists in the 3D world.</td>
<td></td>
</tr>
<tr>
<td>ShapeBase</td>
<td>A scriptable, renderable shape.</td>
<td></td>
</tr>
<tr>
<td>ShapeBaseData</td>
<td>Defines properties for a ShapeBase object.</td>
<td></td>
</tr>
<tr>
<td>ShapeBaseImageData</td>
<td>Represents geometry to be mounted to a ShapeBase object.</td>
<td></td>
</tr>
<tr>
<td>SpawnSphere</td>
<td>This class is used for creating any type of game object, assigning it a class, datablock, and other properties when it is spawned.</td>
<td></td>
</tr>
<tr>
<td>StaticShape</td>
<td>The most basic 3D shape with a datablock available in Torque 3D.</td>
<td></td>
</tr>
<tr>
<td>StaticShapeData</td>
<td>The most basic ShapeBaseData derived shape datablock available in Torque 3D.</td>
<td></td>
</tr>
<tr>
<td>Trigger</td>
<td>A Trigger is a volume of space that initiates script callbacks when objects pass through the Trigger.</td>
<td></td>
</tr>
<tr>
<td>TriggerData</td>
<td>Defines shared properties for Trigger objects.</td>
<td></td>
</tr>
<tr>
<td>TSShapeConstructor</td>
<td>An object used to modify a DTS or COLLADA shape model after it has been loaded by Torque.</td>
<td></td>
</tr>
<tr>
<td>class</td>
<td>TSStatic</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>---------------------------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>A static object derived from a 3D model file and placed within the game world. More...</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th>TurretShape</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Base turret class. More...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th>TurretShapeData</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Defines properties for a TurretShape object. More...</td>
</tr>
</tbody>
</table>
## Enumerations

<table>
<thead>
<tr>
<th>Enum</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ItemLightType</code></td>
<td>The type of light the <strong>Item</strong> has.</td>
</tr>
<tr>
<td><code>PlayerPose</code></td>
<td>The pose of the <strong>Player</strong>.</td>
</tr>
<tr>
<td><code>ShapeBaseImageLightType</code></td>
<td>The type of light to attach to this <strong>ShapeBaseImage</strong>.</td>
</tr>
<tr>
<td><code>ShapeBaseImageLoadedState</code></td>
<td></td>
</tr>
</tbody>
</table>

### `ItemLightType` enum
```csharp
enum ItemLightType {
    NoLight,  // No light
    ConstantLight,
    PulsingLight
}
```

### `PlayerPose` enum
```csharp
enum PlayerPose {
    Stand,    // Standing
    Sprint,   // Sprinting
    Crouch,   // Crouching
    Prone,    // Prone
    Swim      // Swimming
}
```

### `ShapeBaseImageLightType` enum
```csharp
enum ShapeBaseImageLightType {
    NoLight,     // No light
    ConstantLight,
    SpotLight,   // Spot light
    PulsingLight,
    WeaponFireLight
}
```

### `ShapeBaseImageLoadedState` enum
```csharp
enum ShapeBaseImageLoadedState {
    Ignore,     // Ignore
    Loaded,     // Loaded
    Empty       // Empty
}
```
The loaded state of this ShapeBaseImage.

More...

enum ShapeBaseImageRecoilState {
  NoRecoil,
  LightRecoil,
  MediumRecoil,
  HeavyRecoil
}

What kind of recoil this ShapeBaseImage should emit when fired.

More...

enum ShapeBaseImageSpinState {
  Ignore,
  Stop,
  SpinUp,
  SpinDown,
  FullSpeed
}

How the spin animation should be played.

More...

enum TSMeshType {
  None,
  Bounds,
  Mesh
}

Type of mesh data available in a shape.

More...
enum TurretShapeFireLinkType {
    FireTogether,
    GroupedFire,
    IndividualFire
}

How the weapons are linked to triggers for this TurretShape.

More...
# Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float $SB::CloakSpeed</td>
<td>Time to cloak, in seconds.</td>
</tr>
<tr>
<td>float $SB::DFDec</td>
<td>Speed to reduce the damage flash effect per tick.</td>
</tr>
<tr>
<td>float $SB::FullCorrectionDistance</td>
<td>Distance at which a weapon's muzzle vector is fully corrected to match where the player is looking.</td>
</tr>
<tr>
<td>static bool Trigger::renderTriggers</td>
<td>Forces all Trigger's to render.</td>
</tr>
<tr>
<td>float $SB::WODec</td>
<td>Speed to reduce the whiteout effect per tick.</td>
</tr>
</tbody>
</table>
Detailed Description

Objects which can be controlled or directly interact with a user, such as Player, Projectile, Item, etc.

Does not include vehicles as they have their own section.
Enumeration Type Documentation

**enum ItemLightType**

The type of light the **Item** has.

**Enumerator:**

- *NoLight*  
  The item has no light attached.

- *ConstantLight*  
  The item has a constantly emitting light attached.

- *PulsingLight*  
  The item has a pulsing light attached.

**enum PlayerPose**

The pose of the **Player**.

**Enumerator:**

- *Stand*  
  Standard movement pose.

- *Sprint*  
  Sprinting pose.

- *Crouch*  
  Crouch pose.

- *Prone*  
  Prone pose.

- *Swim*  
  Swimming pose.
enum ShapeBaseImageLightType

The type of light to attach to this ShapeBaseImage.

**Enumerator:**

- **NoLight**  No light is attached.
- **ConstantLight**  A constant emitting light is attached.
- **SpotLight**  A spotlight is attached.
- **PulsingLight**  A pulsing light is attached.
- **WeaponFireLight**  Light emits when the weapon is fired, then dissipates.

enum ShapeBaseImageLoadedState

The loaded state of this ShapeBaseImage.

**Enumerator:**

- **Ignore**  Ignore the loaded state.
- **Loaded**  ShapeBaseImage is loaded.
- **Empty**  ShapeBaseImage is not loaded.
enum ShapeBaseImageRecoilState

What kind of recoil this ShapeBaseImage should emit when fired.

**Enumerator:**

- **NoRecoil**  
  No recoil occurs.

- **LightRecoil**  
  A light recoil occurs.

- **MediumRecoil**  
  A medium recoil occurs.

- **HeavyRecoil**  
  A heavy recoil occurs.

enum ShapeBaseImageSpinState

How the spin animation should be played.

**Enumerator:**

- **Ignore**  
  No changes to the spin sequence.

- **Stop**  
  Stops the spin sequence at its current position.

- **SpinUp**  
  Increase spin sequence timeScale from 0 (on state entry) to 1 (after stateTimeoutValue seconds).

  Decrease spin sequence timeScale from 1 (on
**SpinDown** state entry) to 0 (after stateTimeoutValue seconds).

**FullSpeed** Resume the spin sequence playback at its current position with timeScale = 1.

```plaintext
enum TSMeshType

Type of mesh data available in a shape.

**Enumerator:**

* None    No mesh data.

* Bounds  Bounding box of the shape.

* Mesh    Specifically desiganted "collision" meshes.

* Mesh    Rendered mesh polygons.
```

```plaintext
enum TurretShapeFireLinkType

How the weapons are linked to triggers for this TurretShape.

**Enumerator:**

* FireTogether  All weapons fire under trigger 0.

* GroupedFire  Weapon mounts 0,2 fire under trigger 0, mounts 1,3 fire under trigger 1.
IndividualFire  Each weapon mount fires under its own trigger 0-3.
Variable Documentation

float $SB::CloakSpeed

Time to cloak, in seconds.

float $SB::DFDec

Speed to reduce the damage flash effect per tick.

See also:
- ShapeBase::setDamageFlash()
- ShapeBase::getDamageFlash()

Note:
- Relies on the flash postFx.

float $SB::FullCorrectionDistance

Distance at which a weapon's muzzle vector is fully corrected to match where the player is looking.

When a weapon image has correctMuzzleVector set and the Player is in 1st person, the muzzle vector from the weapon is modified to match where the player is looking. Beyond the FullCorrectionDistance the muzzle vector is always corrected. Between FullCorrectionDistance and the player, the weapon's muzzle vector is adjusted so that the closer the aim point is to the player, the closer the muzzle vector is to the true (non-corrected) one.

bool Trigger::renderTriggers [static, inherited]
Forces all Trigger's to render.

Used by the Tools and debug render modes.

| float $SB::WODec |

Speed to reduce the whiteout effect per tick.

**See also:**
- ShapeBase::setWhiteOut()
- ShapeBase::getWhiteOut

**Note:**
- Relies on the flash postFx.

Copyright © GarageGames, LLC. All Rights Reserved.
Miscellaneous
[Game]

Miscellaneous/uncategorized game related objects. More...

Miscellaneous/uncategorized game related objects.

Copyright © GarageGames, LLC. All Rights Reserved.
Physics
[Game]

Objects and functions related to Torque 3D's physics layer. More...
### Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PhysicsDebris</strong></td>
<td>Represents one or more rigid bodies defined in a single mesh file with a limited lifetime. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>PhysicsDebrisData</strong></td>
<td>Defines the properties of a PhysicsDebris object. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>PhysicsForce</strong></td>
<td>Helper object for gameplay physical forces. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>PhysicsShape</strong></td>
<td>Represents a destructible physical object simulated through the plugin system. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>PhysicsShapeData</strong></td>
<td>Defines the properties of a PhysicsShape. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>PxCloth</strong></td>
<td>Rectangular patch of cloth simulated by PhysX. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>PxMaterial</strong></td>
<td>Defines a PhysX material assignable to a PxMaterial. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>PxMultiActor</strong></td>
<td>Represents a destructible physical object simulated using PhysX. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>PxMultiActorData</strong></td>
<td>Defines the properties of a type of PxMultiActor. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>RadialImpulseEvent</strong></td>
<td>Creates a physics-based impulse effect from a defined central point and magnitude. <a href="#">More...</a></td>
</tr>
</tbody>
</table>
Enumerations

```c
enum PhysicsSimType {
    ClientOnly,
    ServerOnly,
    ClientServer
}
```

How to handle the physics simulation with the client's and server.

More...
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>$\text{PhysXLogWarnings}$</td>
<td>Output PhysX warnings to the console.</td>
</tr>
<tr>
<td>bool</td>
<td>$\text{Physics::isSinglePlayer}$</td>
<td>Informs the physics simulation if only a single player exists.</td>
</tr>
<tr>
<td>bool</td>
<td>$\text{physicsPluginPresent}$</td>
<td>Returns true if a physics plugin exists and is initialized.</td>
</tr>
<tr>
<td>int</td>
<td>$\text{pref::Physics::threadCount}$</td>
<td>Number of threads to use in a single pass of the physics engine.</td>
</tr>
</tbody>
</table>
Detailed Description

Objects and functions related to Torque 3D's physics layer.
**Enumeration Type Documentation**

```csharp
enum PhysicsSimType
```

How to handle the physics simulation with the client's and server.

**Enumerator:**

- **ClientOnly**  Only handle physics on the client.
- **ServerOnly**  Only handle physics on the server.
- **ClientServer** Handle physics on both the client and server.
Variable Documentation

bool $PhysXLogWarnings

Output PhysX warnings to the console.

bool $Physics::isSinglePlayer

Informs the physics simulation if only a single player exists.
If true, optimizations will be implemented to better cater to a single player environment.

bool physicsPluginPresent

Returns true if a physics plugin exists and is initialized.

physicsPluginPresent()

int $pref::Physics::threadCount

Number of threads to use in a single pass of the physics engine.
Defaults to 2 if not set.

Copyright © GarageGames, LLC. All Rights Reserved.
Special Effects
[Game]

Classes responsible for special effect objects, such as Explosion, Debris, Particles, etc. More...
# Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debris</td>
<td>Base debris class. Uses the DebrisData datablock for properties of individual debris objects. More...</td>
</tr>
<tr>
<td>DebrisData</td>
<td>Stores properties for an individual debris type. More...</td>
</tr>
<tr>
<td>DecalData</td>
<td>A datablock describing an individual decal. More...</td>
</tr>
<tr>
<td>DecalManager</td>
<td>The object that manages all of the decals in the active mission. More...</td>
</tr>
<tr>
<td>Explosion</td>
<td>The emitter for an explosion effect, with properties defined by a ExplosionData object. More...</td>
</tr>
<tr>
<td>ExplosionData</td>
<td>Defines the attributes of an Explosion: particleEmitters, debris, lighting and camera shake effects. More...</td>
</tr>
<tr>
<td>ForestWindEmitter</td>
<td>Object responsible for simulating wind in a level. More...</td>
</tr>
<tr>
<td>LightAnimData</td>
<td>A datablock which defines and performs light animation, such as rotation, brightness fade, and colorization. More...</td>
</tr>
<tr>
<td>Lightning</td>
<td>An emitter for lightning bolts. More...</td>
</tr>
<tr>
<td>LightningData</td>
<td>Common data for a Lightning emitter object. More...</td>
</tr>
<tr>
<td>LightningStrikeEvent</td>
<td>Network event that triggers a lightning strike on the client when it is received. More...</td>
</tr>
<tr>
<td>ParticleData</td>
<td></td>
</tr>
</tbody>
</table>
Contains information for how specific particles should look and react including particle colors, particle imagemap, acceleration value for individual particles and spin information. More...

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ParticleEmitter</td>
<td>This object is responsible for spawning particles. More...</td>
</tr>
<tr>
<td>ParticleEmitterData</td>
<td>Defines particle emission properties such as ejection angle, period and velocity for a ParticleEmitter. More...</td>
</tr>
<tr>
<td>ParticleEmitterNode</td>
<td>A particle emitter object that can be positioned in the world and dynamically enabled or disabled. More...</td>
</tr>
<tr>
<td>ParticleEmitterNodeData</td>
<td>Contains additional data to be associated with a ParticleEmitterNode. More...</td>
</tr>
<tr>
<td>Precipitation</td>
<td>Defines a precipitation based storm (rain, snow, etc). More...</td>
</tr>
<tr>
<td>PrecipitationData</td>
<td>Defines the droplets used in a storm (raindrops, snowflakes, etc). More...</td>
</tr>
<tr>
<td>Splash</td>
<td>Manages the ring used for a Splash effect. More...</td>
</tr>
<tr>
<td>SplashData</td>
<td>Acts as the physical point in space in white a Splash is created from. More...</td>
</tr>
</tbody>
</table>
Modules

Decals

Decals are non-SimObject derived objects that are stored and loaded separately from the normal mission file.
## Enumerations

```cpp
enum ParticleBlendStyle {
    NORMAL,
    ADDITIVE,
    SUBTRACTION,
    PREMULTALPHA
}
```

The type of visual blending style to apply to the particles.

More...
Functions

float calcExplosionCoverage (Point3F pos=Point3F(0.0f, 0.0f, 0.0f),
int id=NULL, int covMask=NULL)

Calculates how much an explosion effects a specific object.
Detailed Description

Classes responsible for special effect objects, such as Explosion, Debris, Particles, etc.
Enumeration Type Documentation

enum ParticleBlendStyle

The type of visual blending style to apply to the particles.

Enumerator:

NORMAL No blending style.

ADDITIVE Adds the color of the pixel to the frame buffer with full alpha for each pixel.

SUBTRACTIVE Subtractive Blending. Reverses the color model, causing dark colors to have a stronger visual effect.

PREMULTALPHA Color blends with the colors of the imagemap rather than the alpha.
Function Documentation

```cpp
float calcExplosionCoverage(Point3F pos = Point3F(0.0f, 0.0f, 0.0f),
                            int    id = NULL,
                            int    covMask = NULL)
```

Calculates how much an explosion effects a specific object.

Use this to determine how much damage to apply to objects based on their distance from the explosion's center point, and whether the explosion is blocked by other objects.

**Parameters:**
- `pos` Center position of the explosion.
- `id` Id of the object of which to check coverage.
- `covMask` Mask of object types that may block the explosion.

**Returns:**
Coverage value from 0 (not affected by the explosion) to 1 (fully affected)

**Example:**

```cpp
// Get the position of the explosion.
%position = %explosion.getPosition();

// Set a list of TypeMasks (defined in gameFunctioncs.cpp), separated by the | character.
%TypeMasks = $TypeMasks::StaticObjectType | $TypeMasks::ItemObjectType

// Acquire the damage value from 0.0f - 1.
%coverage = calcExplosionCoverage( %position,
                                  %sceneObject,
                                  %TypeMasks);

// Apply damage to object
%sceneObject.applyDamage( %coverage * 20 );
```
Decals
[Special Effects]

Decals are non-SimObject derived objects that are stored and loaded separately from the normal mission file. More...
## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DecalData</strong></td>
<td>A datablock describing an individual decal. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>DecalManager</strong></td>
<td>The object that manages all of the decals in the active mission. <a href="#">More...</a></td>
</tr>
</tbody>
</table>
## Functions

<table>
<thead>
<tr>
<th>Function Type</th>
<th>Function Name</th>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><code>decalManagerAddDecal</code></td>
<td><code>(Point3F position, Point3F normal, float rot, float scale, DecalData decalData, bool isImmortal=false)</code></td>
<td>Adds a new decal to the decal manager.</td>
</tr>
<tr>
<td>void</td>
<td><code>decalManagerClear</code></td>
<td></td>
<td>Removes all decals currently loaded in the decal manager.</td>
</tr>
<tr>
<td>bool</td>
<td><code>decalManagerDirty</code></td>
<td></td>
<td>Returns whether the decal manager has unsaved modifications.</td>
</tr>
<tr>
<td>bool</td>
<td><code>decalManagerLoad</code></td>
<td><code>(string fileName)</code></td>
<td>Clears existing decals and replaces them with decals loaded from the specified file.</td>
</tr>
<tr>
<td>bool</td>
<td><code>decalManagerRemoveDecal</code></td>
<td><code>(int decalID)</code></td>
<td>Remove specified decal from the scene.</td>
</tr>
<tr>
<td>void</td>
<td><code>decalManagerSave</code></td>
<td><code>(String decalSaveFile=&quot;&quot;)</code></td>
<td>Saves the decals for the active mission in the entered filename.</td>
</tr>
</tbody>
</table>
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>$Decals::debugRender</td>
<td>If true, the decal spheres will be visualized when in the editor.</td>
</tr>
<tr>
<td>bool</td>
<td>$pref::Decals::enabled</td>
<td>Controls whether decals are rendered.</td>
</tr>
<tr>
<td>float</td>
<td>$pref::Decals::lifeTimeScale</td>
<td>Lifetime that decals will last after being created in the world. Deprecated. Use DecalData::lifeSpan instead.</td>
</tr>
<tr>
<td>bool</td>
<td>$Decals::poolBuffers</td>
<td>If true, will merge all PrimitiveBuffers and VertexBuffers into a pair of pools before clearing them at the end of a frame.</td>
</tr>
<tr>
<td>float</td>
<td>$Decals::sphereDistanceTolerance</td>
<td>The distance at which the decal system will start breaking up decal spheres when adding new decals.</td>
</tr>
<tr>
<td>float</td>
<td>$Decals::sphereRadiusTolerance</td>
<td>The radius beyond which the decal system will start breaking up decal spheres when adding new decals.</td>
</tr>
</tbody>
</table>
Detailed Description

Decals are non-SimObject derived objects that are stored and loaded separately from the normal mission file.

The DecalManager handles all aspects of decal management including loading, creation, saving, and automatically deleting decals that have exceeded their lifeSpan.

The static decals associated with a mission are normally loaded immediately after the mission itself has loaded as shown below.

Example:

```// Load the static mission decals.
decalManagerLoad( %missionName @ ".decals"
```
Function Documentation

```c
int decalManagerAddDecal(Point3F position,  
Point3F normal,  
float rot,  
float scale,  
DecalData decalData,  
bool isImmortal = false
)
```

Adds a new decal to the decal manager.

**Parameters:**

- `position`  World position for the decal.
- `normal`    Decal normal vector (if the decal was a tire lying flat on a surface, this is the vector pointing in the direction of the axle).
- `rot`       Angle (in radians) to rotate this decal around its normal vector.
- `scale`     Scale factor applied to the decal.
- `decalData` DecalData datablock to use for the new decal.
- `isImmortal` Whether or not this decal is immortal. If immortal, it does not expire automatically and must be removed explicitly.

**Returns:**

Returns the ID of the new Decal object or -1 on failure.

**Example:**

```c
// Specify the decal position
%position = "1.0 1.0 1.0";

// Specify the up vector
%normal = "0.0 0.0 1.0";
```
// Add the new decal.
%decalObj = decalManagerAddDecal( %position, %normal, 0.5, 0.35, ScorchBigDecal,
void decalManagerClear( )

Removes all decals currently loaded in the decal manager.

Example:

// Tell the decal manager to remove all existing decals.
decalManagerClear();

bool decalManagerDirty( )

Returns whether the decal manager has unsaved modifications.

Returns:
True if the decal manager has unsaved modifications, false if everything has been saved.

Example:

// Ask the decal manager if it has unsaved modifications.
%hasUnsavedModifications = decalManagerDirty();

bool decalManagerLoad(string fileName )

Clears existing decals and replaces them with decals loaded from the specified file.

Parameters:

fileName Filename to load the decals from.

Returns:
True if the decal manager was able to load the requested file, false if it could not.

Example:

```c
// Set the filename to load the decals from.
%fileName = "./missionDecals.mis.decals";
// Inform the decal manager to load the decals from the entered filename.
decalManagerLoad( %fileName );
```

```c
bool decalManagerRemoveDecal(int decalID )
```

Remove specified decal from the scene.

**Parameters:**

- `decalID` ID of the decal to remove.

**Returns:**

Returns true if successful, false if decal ID not found.

Example:

```c
// Specify a decal ID to be removed
%decalID = 1;

// Tell the decal manager to remove the specified decal ID.
decalManagerRemoveDecal( %decalId )
```

```c
void decalManagerSave(String decalSaveFile = "" )
```

Saves the decals for the active mission in the entered filename.

**Parameters:**

- `decalSaveFile` Filename to save the decals to.
Example:

```plaintext
// Set the filename to save the decals in. 
// decals will default to <activeMissionName>.mis.decals
%fileName = "./missionDecals.mis.decals";
// Inform the decal manager to save the decals
decalManagerSave( %fileName );
```
# Variable Documentation

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bool $Decals::debugRender</code></td>
<td>If true, the decal spheres will be visualized when in the editor.</td>
</tr>
<tr>
<td><code>bool $pref::Decals::enabled</code></td>
<td>Controls whether decals are rendered.</td>
</tr>
<tr>
<td><code>float $pref::Decals::lifeTimeScale</code></td>
<td>Lifetime that decals will last after being created in the world. Deprecated. Use <code>DecalData::lifeSpan</code> instead.</td>
</tr>
<tr>
<td><code>bool $Decals::poolBuffers</code></td>
<td>If true, will merge all PrimitiveBuffers and VertexBuffers into a pair of pools before clearing them at the end of a frame. If false, will just clear them at the end of a frame.</td>
</tr>
<tr>
<td><code>float $Decals::sphereDistanceTolerance</code></td>
<td>The distance at which the decal system will start breaking up decal spheres when adding new decals.</td>
</tr>
<tr>
<td><code>float $Decals::sphereRadiusTolerance</code></td>
<td>The radius beyond which the decal system will start breaking up...</td>
</tr>
</tbody>
</table>
decal spheres when adding new decals.
Vehicles
[Game]

This section is dedicated to vehicle game objects, such as the base Vehicle class, WheeledVehicle, FlyingVehicle, and so on. More...
## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FlyingVehicle</td>
<td>A flying vehicle. <a href="#">More...</a></td>
</tr>
<tr>
<td>FlyingVehicleData</td>
<td>Defines the properties of a FlyingVehicle. <a href="#">More...</a></td>
</tr>
<tr>
<td>HoverVehicle</td>
<td>A hovering vehicle. <a href="#">More...</a></td>
</tr>
<tr>
<td>HoverVehicleData</td>
<td>Defines the properties of a HoverVehicle. <a href="#">More...</a></td>
</tr>
<tr>
<td>WheeledVehicle</td>
<td>A wheeled vehicle. <a href="#">More...</a></td>
</tr>
<tr>
<td>WheeledVehicleData</td>
<td>Defines the properties of a WheeledVehicle. <a href="#">More...</a></td>
</tr>
<tr>
<td>WheeledVehicleSpring</td>
<td>Defines the properties of a WheeledVehicle spring. <a href="#">More...</a></td>
</tr>
<tr>
<td>WheeledVehicleTire</td>
<td>Defines the properties of a WheeledVehicle tire. <a href="#">More...</a></td>
</tr>
</tbody>
</table>
Detailed Description

This section is dedicated to vehicle game objects, such as the base Vehicle class, WheeledVehicle, FlyingVehicle, and so on.
Input Management

Functions and classes relating to user input. More...
<table>
<thead>
<tr>
<th>class</th>
<th>ActionMap</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ActionMaps assign platform input events to console commands.</td>
</tr>
</tbody>
</table>
## Modules

**Input Event Listing**

List of all possible input events that can be mapped.
## Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td><code>activateDirectInput()</code></td>
<td>Activates DirectInput.</td>
</tr>
<tr>
<td>void</td>
<td><code>deactivateDirectInput()</code></td>
<td>Disables DirectInput.</td>
</tr>
<tr>
<td>void</td>
<td><code>disableJoystick()</code></td>
<td>Disables use of the joystick.</td>
</tr>
<tr>
<td>void</td>
<td><code>disableXInput()</code></td>
<td>Disables XInput for Xbox 360 controllers.</td>
</tr>
<tr>
<td>void</td>
<td><code>echoInputState()</code></td>
<td>Prints information to the console stating if DirectInput and a Joystick are enabled and active.</td>
</tr>
<tr>
<td>bool</td>
<td><code>enableJoystick()</code></td>
<td>Enables use of the joystick.</td>
</tr>
<tr>
<td>bool</td>
<td><code>enableXInput()</code></td>
<td>Enables XInput for Xbox 360 controllers.</td>
</tr>
<tr>
<td>ActionMap</td>
<td><code>getCurrentActionMap()</code></td>
<td>Returns the current ActionMap.</td>
</tr>
<tr>
<td>int</td>
<td><code>getXInputState(int controllerID, string property, bool current)</code></td>
<td>Queries the current state of a connected Xbox 360 controller.</td>
</tr>
<tr>
<td>bool</td>
<td><code>isJoystickEnabled()</code></td>
<td>Queries input manager to see if a joystick is enabled.</td>
</tr>
<tr>
<td>bool</td>
<td><code>isXInputConnected(int controllerID)</code></td>
<td>Checks to see if an Xbox 360 controller is connected.</td>
</tr>
<tr>
<td>void</td>
<td><code>lockMouse(bool isLocked)</code></td>
<td>Lock or unlock the mouse to the window.</td>
</tr>
<tr>
<td>void</td>
<td><code>resetXInput()</code></td>
<td>Rebuilds the XInput section of the InputManager.</td>
</tr>
</tbody>
</table>
void  **rumble**  (string device, float xRumble, float yRumble)

Activates the vibration motors in the specified controller.
Detailed Description

Functions and classes relating to user input.

Input events come from the OS, are translated in the platform layer and then posted to the game. By default the game then checks the input event against a global ActionMap (which supercedes all other action handlers). If there is no action specified for the event, it is passed on to the GUI system. If the GUI does not handle the input event it is passed to the currently active (non-global) ActionMap stack.

Example: the user presses the ~ (tilde) key, which is bound in the global ActionMap to toggleConsole.

This causes the console function associated with the bind to be executed, which in this case is toggleConsole, resulting in the console output window being shown. If the key had not been bound in the global map, it would have passed to the first gui that could have handled it, and if none did, it would pass to any game actions that were bound to that key.

See also:

ActionMap
## Function Documentation

### void activateDirectInput( )

Activates DirectInput.
Also activates any connected joysticks.

### void deactivateDirectInput( )

Disables DirectInput.
Also deactivates any connected joysticks.

### void disableJoystick( )

Disables use of the joystick.

**Note:**
DirectInput must be enabled and active to use this function.

### void disableXInput( )

Disables XInput for Xbox 360 controllers.

### void echoInputState( )

Prints information to the console stating if DirectInput and a Joystick are enabled and active.

### bool enableJoystick( )
Enables use of the joystick.

**Note:**
DirectInput must be enabled and active to use this function.

```c
bool enableXInput()
```

Enables XInput for Xbox 360 controllers.

**Note:**
XInput is enabled by default. Disable to use an Xbox 360 Controller as a joystick device.

```c
ActionMap getCurrentActionMap()
```

Returns the current ActionMap.

**See also:**
ActionMap

```c
int getXInputState(int controllerID,
                    string property,
                    bool current
)
```

Queries the current state of a connected Xbox 360 controller.

**XInput Properties:**

- `XI_THUMBLX, XI_THUMBLY` - X and Y axes of the left thumbstick.
- `XI_THUMBRX, XI_THUMBRY` - X and Y axes of the right thumbstick.
- **XI_LEFT_TRIGGER, XI_RIGHT_TRIGGER** - Left and Right triggers.
- **SI_UPOV, SI_DPOV, SI_LPOV, SI_RPOV** - Up, Down, Left, and Right on the directional pad.
- **XI_START, XI_BACK** - The Start and Back buttons.
- **XI_LEFT_THUMB, XI_RIGHT_THUMB** - Clicking in the left and right thumbstick.
- **XI_LEFT_SHOULDER, XI_RIGHT_SHOULDER** - Left and Right bumpers.

**Parameters:**
- `controllerID` - Zero-based index of the controller to return information about.
- `property` - Name of input action being queried, such as "XI_THUMBLX".
- `current` - True checks current device in action.

**Returns:**
- Button queried - 1 if the button is pressed, 0 if it's not.
- Thumbstick queried - Int representing displacement from rest position.
- Trigger queried - Int from 0 to 255 representing how far the trigger is displaced.

```c
bool isJoystickEnabled( )
```

Queries input manager to see if a joystick is enabled.

**Returns:**
- 1 if a joystick exists and is enabled, 0 if it's not.

```c
bool isXInputConnected(int controllerID )
```

Checks to see if an Xbox 360 controller is connected.
Parameters:

`controllerID` Zero-based index of the controller to check.

Returns:

1 if the controller is connected, 0 if it isn't, and 205 if XInput hasn't been initialized.

```cpp
void lockMouse(bool isLocked )
```

Lock or unlock the mouse to the window.

When true, prevents the mouse from leaving the bounds of the game window.

```cpp
void resetXInput( )
```

Rebuilds the XInput section of the InputManager.

Requests a full refresh of events for all controllers. Useful when called at the beginning of game code after actionMaps are set up to hook up all appropriate events.

```cpp
void rumble(string device,
             float xRumble,
             float yRumble )
```

Activates the vibration motors in the specified controller.

The controller will constantly at it's xRumble and yRumble intensities until changed or told to stop. Valid inputs for xRumble/yRumble are [0 - 1].

Parameters:
device   Name of the device to rumble.
xRumble   Intensity to apply to the left motor.
yRumble   Intensity to apply to the right motor.

Note:
in an Xbox 360 controller, the left motor is low-frequency, while the right motor is high-frequency.
## Input Event Listing

[Input Management]

List of all possible input events that can be mapped. More...

List of all possible input events that can be mapped.

The following table represents all keyboard, mouse, and joystick input events available to stock Torque 3D. It should be noted that letter and number keys directly correlate to their mapping. For example "a" is literally the letter a. The button0, button1, and button2 are the most commonly used input mappings for left mouse button, right mouse button, and middle mouse button (respectively).

<table>
<thead>
<tr>
<th>Keyboard General Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event Name</td>
</tr>
<tr>
<td>backspace</td>
</tr>
<tr>
<td>tab</td>
</tr>
<tr>
<td>return</td>
</tr>
<tr>
<td>enter</td>
</tr>
<tr>
<td>shift</td>
</tr>
</tbody>
</table>

**Note:** All general keys can be bound by simply using the key... ex. "u" will trigger the u key response.
<table>
<thead>
<tr>
<th>ctrl</th>
<th>down</th>
<th>numlock</th>
<th>backslash</th>
</tr>
</thead>
<tbody>
<tr>
<td>alt</td>
<td>print</td>
<td>scrolllock</td>
<td>semicolon</td>
</tr>
<tr>
<td>pause</td>
<td>insert</td>
<td>rshift</td>
<td>apostrophe</td>
</tr>
<tr>
<td>capslock</td>
<td>delete</td>
<td>lcontrol</td>
<td>comma</td>
</tr>
<tr>
<td>escape</td>
<td>help</td>
<td>rcontrol</td>
<td>period</td>
</tr>
<tr>
<td>space</td>
<td>win_lwindow</td>
<td>lalt</td>
<td>slash</td>
</tr>
<tr>
<td>pagedown</td>
<td>win_rwindow</td>
<td>ralt</td>
<td>lessthan</td>
</tr>
<tr>
<td>pageup</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Keyboard Numpad Events**

<table>
<thead>
<tr>
<th>Event Name</th>
<th>Event Name</th>
<th>Event Name</th>
<th>Event Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>numpad0</td>
<td>numpad5</td>
<td>numpad9</td>
<td>numpadadminus</td>
</tr>
<tr>
<td>numpad1</td>
<td>numpad6</td>
<td>numpadadmult</td>
<td>numpaddecimal</td>
</tr>
<tr>
<td>numpad2</td>
<td>numpad7</td>
<td>numpadadd</td>
<td>numpaddivide</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>-----------</td>
<td>--------------</td>
</tr>
<tr>
<td>numpad3</td>
<td>numpad8</td>
<td>numpadsep</td>
<td>numpadenter</td>
</tr>
<tr>
<td>numpad4</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Keyboard Function Key Events**

<table>
<thead>
<tr>
<th>Event Name</th>
<th>Event Name</th>
<th>Event Name</th>
<th>Event Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>f1</td>
<td>f7</td>
<td>f13</td>
<td>f19</td>
</tr>
<tr>
<td>f2</td>
<td>f8</td>
<td>f14</td>
<td>f20</td>
</tr>
<tr>
<td>f3</td>
<td>f9</td>
<td>f15</td>
<td>f21</td>
</tr>
<tr>
<td>f4</td>
<td>f10</td>
<td>f16</td>
<td>f22</td>
</tr>
<tr>
<td>f5</td>
<td>f11</td>
<td>f17</td>
<td>f23</td>
</tr>
<tr>
<td>f6</td>
<td>f12</td>
<td>f18</td>
<td>f24</td>
</tr>
<tr>
<td>Event Name</td>
<td>Event Name</td>
<td>Event Name</td>
<td>Event Name</td>
</tr>
<tr>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
</tr>
<tr>
<td>button0</td>
<td>button8</td>
<td>button16</td>
<td>button24</td>
</tr>
<tr>
<td>button1</td>
<td>button9</td>
<td>button17</td>
<td>button25</td>
</tr>
<tr>
<td>button2</td>
<td>button10</td>
<td>button18</td>
<td>button26</td>
</tr>
<tr>
<td>button3</td>
<td>button11</td>
<td>button19</td>
<td>button27</td>
</tr>
<tr>
<td>button4</td>
<td>button12</td>
<td>button20</td>
<td>button28</td>
</tr>
<tr>
<td>button5</td>
<td>button13</td>
<td>button21</td>
<td>button29</td>
</tr>
<tr>
<td>button6</td>
<td>button14</td>
<td>button22</td>
<td>button30</td>
</tr>
<tr>
<td>button7</td>
<td>button15</td>
<td>button23</td>
<td>button31</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Event Name</th>
<th>Event Name</th>
<th>Event Name</th>
<th>Event Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>xaxis</td>
<td>zaxis</td>
<td>ryaxis</td>
<td>slider</td>
</tr>
<tr>
<td>Event Name</td>
<td>Event Name</td>
<td>Event Name</td>
<td>Event Name</td>
</tr>
<tr>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
</tr>
<tr>
<td>x pov</td>
<td>d pov</td>
<td>x pov2</td>
<td>d pov2</td>
</tr>
<tr>
<td>y pov</td>
<td>l pov</td>
<td>y pov2</td>
<td>l pov2</td>
</tr>
<tr>
<td>u pov</td>
<td>r pov</td>
<td>u pov2</td>
<td>r pov2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Event Name</th>
<th>Event Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>anykey</td>
<td>nomatch</td>
</tr>
</tbody>
</table>

See also:
- ActionMap
Networking

Classes and functions related Torque 3D networking. More...
<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIClient</td>
<td>Simulated client driven by AI commands. <a href="#">More...</a></td>
</tr>
<tr>
<td>AIConnection</td>
<td>Special client connection driven by an AI, rather than a human. <a href="#">More...</a></td>
</tr>
<tr>
<td>GameConnection</td>
<td>The game-specific subclass of <a href="#">NetConnection</a>. <a href="#">More...</a></td>
</tr>
<tr>
<td>HTTPObject</td>
<td>Allows communications between the game and a server using HTTP. <a href="#">More...</a></td>
</tr>
<tr>
<td>NetConnection</td>
<td>Provides the basis for implementing a multiplayer game protocol. <a href="#">More...</a></td>
</tr>
<tr>
<td>NetObject</td>
<td>Superclass for all ghostable networked objects. <a href="#">More...</a></td>
</tr>
<tr>
<td>SimpleMessageEvent</td>
<td>A very simple example of a network event. <a href="#">More...</a></td>
</tr>
<tr>
<td>SimpleNetObject</td>
<td>A very simple example of a class derived from <a href="#">NetObject</a>. <a href="#">More...</a></td>
</tr>
<tr>
<td>TCPObject</td>
<td>Allows communications between the game and a server using TCP/IP protocols. <a href="#">More...</a></td>
</tr>
</tbody>
</table>
## Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>addTaggedString (string str)</code></td>
<td>Use the addTaggedString function to tag a new string and add it to the NetStringTable.</td>
</tr>
<tr>
<td><code>buildTaggedString (string format,...)</code></td>
<td>Build a string using the specified tagged string format.</td>
</tr>
<tr>
<td><code>closeNetPort ()</code></td>
<td>Closes the current network port.</td>
</tr>
<tr>
<td><code>commandToClient (NetConnection client, string func,...)</code></td>
<td>Send a command from the server to the client.</td>
</tr>
<tr>
<td><code>commandToServer (string func,...)</code></td>
<td>Send a command to the server.</td>
</tr>
<tr>
<td><code>detag (string str)</code></td>
<td>Returns the string from a tag string.</td>
</tr>
<tr>
<td><code>DNetSetLogging (bool enabled)</code></td>
<td>Enables logging of the connection protocols.</td>
</tr>
<tr>
<td><code>dumpNetStats ()</code></td>
<td>Dumps network statistics for each class to the console.</td>
</tr>
<tr>
<td><code>dumpNetStringTable ()</code></td>
<td>Dump the current contents of the networked string table to the console.</td>
</tr>
<tr>
<td><code>getTag (string textTagString)</code></td>
<td>Extracts the tag from a tagged string.</td>
</tr>
<tr>
<td><code>getTaggedString (int tag)</code></td>
<td>Use the getTaggedString function to convert a tag to a string.</td>
</tr>
<tr>
<td><code>pathOnMissionLoadDone ()</code></td>
<td>Load all Path information from the mission.</td>
</tr>
<tr>
<td><code>removeTaggedString (int tag)</code></td>
<td>Remove a tagged string from the Net String Table.</td>
</tr>
</tbody>
</table>
bool setNetPort (int port, bool bind=true)

Set the network port for the game to use.
### Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>void allowConnections</td>
<td>Sets whether or not the global NetInterface allows connections from remote hosts.</td>
</tr>
<tr>
<td>int</td>
<td>$pref::Net::LagThreshold</td>
<td>How long between received packets before the client is considered as lagging (in ms).</td>
</tr>
<tr>
<td>int</td>
<td>$Stats::netBitsReceived</td>
<td>The number of bytes received during the last packet process operation.</td>
</tr>
<tr>
<td>int</td>
<td>$Stats::netBitsSent</td>
<td>The number of bytes sent during the last packet send operation.</td>
</tr>
<tr>
<td>int</td>
<td>$Stats::netGhostUpdates</td>
<td>The total number of ghosts added, removed, and/or updated on the client during the last packet process operation.</td>
</tr>
<tr>
<td>int</td>
<td>$pref::Net::PacketRateToClient</td>
<td>Sets how often packets are sent from the server to a client.</td>
</tr>
<tr>
<td>int</td>
<td>$pref::Net::PacketRateToServer</td>
<td>Sets how often packets are sent from the client to the server.</td>
</tr>
<tr>
<td>int</td>
<td>$pref::Net::PacketSize</td>
<td>Sets the maximum size in bytes an individual network packet may be.</td>
</tr>
</tbody>
</table>
## Callbacks

**void** `onDataBlockObjectReceived` *(int index, int total)*  
Called on the client each time a datablock has been received.

**void** `onLightManagerActivate` *(string name)*  
A callback called by the engine when a light manager is activated.

**void** `onLightManagerDeactivate` *(string name)*  
A callback called by the engine when a light manager is deactivated.
Detailed Description

Classes and functions related Torque 3D networking.

Overview

Torque was designed from the foundation up to offer robust client/server networked simulations.

Performance over the internet drove the design for the networking model. Torque attempts to deal with three fundamental problems of network simulation programming: limited bandwidth, packet loss and latency.

An instance of Torque can be set up as a dedicated server, a client, or both a client and a server. If the game is a client and a server, it still behaves as a client connected to a server - instead of using the network, however, the NetConnection object has a short-circuit link to another NetConnection object in the same application instance. This is known as a local connection.

Handling Limited Bandwidth

Bandwidth is a problem because in the large, open environments
that Torque allows, and with the large number of clients that the engine supports (depending on amount of data sent per client, game world complexity, and available bandwidth), potentially many different objects can be moving and updating at once.

Torque uses three main strategies to maximize available bandwidth. First, it prioritizes data, sending updates to what is most "important" to a client at a greater frequency than it updates data that is less important. Second, it sends only data that is necessary. Using the BitStream class, only the absolute minimum number of bits needed for a given piece of data will be sent. Also, when object state changes, Torque only sends the part of the object state that changed. Last, Torque caches common strings (NetStringTable) and data (SimDataBlock) so that they only need to be transmitted once.

Handling Packet Loss

Packet loss is a problem because the information in lost data packets must somehow be retransmitted, yet in many cases the data in the dropped packet, if resent directly, will be stale by the time it gets to the client.

For example, suppose that packet 2 contains a position update for a player and packet 3 contains a more recent position update for that same player. If packet 2 is dropped but packet 3 makes it across to the client, the engine shouldn't resend the data that was in packet 2. It is older than the version that was received by the client. In order to minimize data that gets resent unnecessarily, the engine classifies

![Diagram of packet loss](image-url)
data into five groups:

- **Unguaranteed Data** (NetEvent) - If this data is lost, don't re-transmit it. An example of this type of data could be real-time voice traffic. By the time it is resent subsequent voice segments will already have played.

- **Guaranteed Data** (NetEvent) - If this data is lost, resend it. This is good for important, one-time information, like which team the player is on, or mission end messages are all examples of guaranteed data.

- **Guaranteed Ordered Data** (NetEvent) - If this data is lost, not only resend it, but make sure it arrives in the correct order. Chat messages, and messages for players joining and leaving the game are all examples of guaranteed, ordered data. In the diagram above, packet 5 arrives before packet 4. If these consist of guaranteed ordered data, the client will not process packet 5 until packet 4 is first handled.

- **Most-Recent State Data** (NetObject) - Only the most current version of the data is important. If an update is lost, send the current state, unless it has been sent already. Most scene objects transmit their information in this manner.

- **Guaranteed Quickest Data** (Move) - Critical data that must get through as soon as possible. An example of this is movement information from the client to the server, which is transmitted with every packet by the Move Manager.

**Handling Latency**

Latency is a problem in the simulation because the network delay in information transfer (which, for modems, can be up to a quarter of a second or more) makes the client's view of the world perpetually out-of-sync with the server.

Twitch FPS games, for which Torque was initially designed, require instant control response in order to feel anything but sluggish. Also, fast moving objects can be difficult for highly latent players to hit. In
order to solve these problems Torque employs several strategies:

- **Interpolation** is used to smoothly move an object from where the client thinks it is to where the server says it is.

- **Extrapolation** is used to guess where the object is going based on its state and rules of movement.

- **Prediction** is used to form an educated guess about where an object is going based on rules of movement and client input.

The network architecture is layered: at the bottom is the platform layer, above that the notify protocol layer, followed by the `NetConnection` object and event management layer.

**On Ghosting and Scoping**

One of the most powerful aspects of Torque's networking code is its support for ghosting and prioritized, most-recent-state network updates.

The way this works is a bit complex, but it is immensely efficient. Let's run through the steps that the server goes through for each client in this part of Torque's networking:

- First, the server determines what objects are in-scope for the client. This is done by calling `onCameraScopeQuery()` on the object which is considered the "scope" object. This is usually the player object, but it can be something else. (For instance, the current vehicle, or an object we're remote controlling.)

- Second, it ghosts them to the client. A ghost is the client's representation of the server's object, and only maintains data that the client requires for the simulation. Ghosts come and go on the client according to the scope rules in the first step above.

- Finally, the server sends updates as needed, by checking the dirty list and packing updates. By only sending dirty data and using bit packing, no excess bandwidth is wasted. The order of ghost updates and their frequency is prioritized by the results of
the object's getUpdatePriority() method.

Each object ghosted is assigned a ghost ID; the client is only aware of the ghost ID. This acts to enhance game security, as it becomes difficult to map objects from one connection to another, or to reliably identify objects from ID alone. IDs are also reassigned based on need, making it hard to track objects that have fallen out of scope (as any object which the player shouldn’t see would).

NetConnection::resolveGhostID() is used on the client side, and NetConnection::resolveObjectFromGhostIndex() on the server side, to turn ghost IDs into object references. NetConnection::getGhostID() is used in the other direction to determine an object's ghost ID based on its SimObject ID.

There is a cap on the maximum number of ghosts per client. Ghost IDs are currently sent via a 12-bit field, ergo, there is a cap of 4096 objects ghosted per client. This can be easily raised; see the GhostConstants enum in the source code.

See also:

NetObject for a further description of ghosting and individual objects.

NetConnection Group

The NetConnection is a SimGroup.

On the client side, it contains all the objects which have been ghosted to that client. On the server side, it is empty. It can be used (typically in script) to hold objects related to the connection. For instance, you might place an observation camera in the NetConnection, or the current Player object. In both cases, when the connection is destroyed, so are the contained objects.

See also:

NetConnection, the basis for implementing a multiplayer game protocol. Also see NetObject, which is the superclass for ghostable objects, and ShapeBase, which is the base for player
Local Connections

It is possible to run both the server and client within the same process.

This is typically done while developing your multiplayer game, and is often required when using Torque's built-in world creation tools. This is also how a single player game is run. Having both a server and client together is known as a local connection.

Any time a player launches the game and chooses to host a mission, they are also making use of a local connection. All other players joining the game use a regular, networked connection, and are considered clients.

Internally, a local connection short-circuits the networking layer and allows for data to pass immediately between the internal server and client. However, it should be noted that there is still the additional overhead of having separate server and client branches within the code, even when creating a single player game. When developing your single player game, you need to be mindful that a client and server still exist within the engine.

See also:

NetConnection, the basis for implementing a multiplayer game protocol.

Monitoring the Network

If you are interested in seeing Torque's various network statistics, use the Net Graph.
The Net Graph is from a client, or ServerConnection, point of view. To activate the Net Graph, either press the 'n' key, or open the console and type 'toggleNetGraph();'. The Net Graph presents a number of networking statistics, as described below:

- **Ghosts Active** The number of active ghosts on the connection.

- **Ghost Updates** The total number of ghosts added, removed or updated since the last update.

- **Bytes Sent** The total number of bytes sent to the server since the last update.

- **Bytes Received** The total number of bytes received from the server since the last update.

- **Latency** The average round trip time (in ms) for the connection. Also known as ping.

- **Packet Loss** The percentage of packets lost since the last update.
Function Documentation

string addTaggedString (string str )

Use the addTaggedString function to tag a new string and add it to the NetStringTable.

Parameters:

str

The string to be tagged and placed in the NetStringTable.

Tagging ignores case, so tagging the same string (excluding case differences) will be ignored as a duplicated tag.

Returns:

Returns a string( containing a numeric value) equivalent to the string ID for the newly tagged string

See also:

Data Types under Tagged Strings
removeTaggedString()
getTaggedString()

string buildTaggedString (string format, ... )

Build a string using the specified tagged string format.

This function takes an already tagged string (passed in as a tagged string ID) and one or more additional strings. If the tagged string contains argument tags that range from %1 through %9, then each additional string will be substituted into the tagged string. The final (non-tagged) combined string will be returned. The maximum length of the tagged string plus any inserted additional strings is 511 characters.
Parameters:

`format` A tagged string ID that contains zero or more argument tags, in the form of %1 through %9. A variable number of arguments that are inserted into the tagged string based on the argument tags within the format string.

Returns:

An ordinary string that is a combination of the original tagged string with any additional strings passed in inserted in place of each argument tag.

Example:

```markdown
// Create a tagged string with argument tags
%taggedStringID = addTaggedString("Welcome

// Some point later, combine the tagged string with some other string
%string = buildTaggedString(%taggedStringID, %playerName);

echo(%string);
```

See also:

- [Data Types](#) under Tagged Strings
- `addTaggedString()`
- `getTaggedString()`

```c
void closeNetPort()
```

Closes the current network port.

```c
void commandToClient(NetConnection client, string string, func func, ...)
```
Send a command from the server to the client.

**Parameters:**

- *client* The numeric ID of a client
- *func* Name of the client function being called
- ... Various parameters being passed to client command

**Example:**

```cpp
// Set up the client command. Needs to be within scripts/client/client.cs
// Update the Ammo Counter with current ammo
function clientCmdSetAmmoAmountHud(%amount)
{
    if (!%amount)
        AmmoAmount.setVisible(false); // Hide the counter
    else
    {
        AmmoAmount.setVisible(true); // Show the counter
        AmmoAmount.setText("Ammo: ", @%amount);
    }
}

// Call it from a server function. Needs such as within scripts/server/game.cs
function GameConnection::setAmmoAmountHud(%client, %amount)
{
    commandToClient(%client, 'SetAmmoAmountHud', @%amount);
}
```

```cpp
void commandToServer(string func,
                    ...
)
```
Send a command to the server.

**Parameters:**

- `func` Name of the server command being called
- ... Various parameters being passed to server command

**Example:**

```cpp
// Create a standard function. Needs to be as within scripts/client/default.bind.cs
function toggleCamera(%val)
{
    // If key was down, call a server command
    if (%val)
        commandToServer('ToggleCamera');
}

// Server command being called from above. Needs to be executed on the server, such as within scripts/server/commands.cs
function serverCmdToggleCamera(%client)
{
    if (%client.getControlObject() == %client.player)
    {
        %client.camera.setVelocity("0 0 0");
        %control = %client.camera;
    }
    else
    {
        %client.player.setVelocity("0 0 0");
        %control = %client.player;
    }
    %client.setControlObject(%control);
    clientCmdSyncEditorGui();
}
```
string detag(string str)

Returns the string from a tag string.

Should only be used within the context of a function that receives a tagged string, and is not meant to be used outside of this context. Use `getTaggedString()` to convert a tagged string ID back into a regular string at any time.

**Example:**

```csharp
// From scripts/client/message.cs
function clientCmdChatMessage(%sender, %voice, %msgString)
{
    onChatMessage(detag(%msgString), %voice);
}
```

See also:
- Data Types under Tagged Strings
- `getPost()`
- `getTaggedString()`

```csharp
void DNetSetLogging(bool enabled)
```

Enables logging of the connection protocols.

When enabled a lot of network debugging information is sent to the console.

**Parameters:**

- `enabled` True to enable, false to disable

```csharp
void dumpNetStats()
```
Dumps network statistics for each class to the console.

The returned *avg*, *min* and *max* values are in bits sent per update. The *num* value is the total number of events collected.

**Note:**

This method only works when TORQUE_NET_STATS is defined in torqueConfig.h.

```c
void dumpNetStringTable()
```

Dump the current contents of the networked string table to the console.

The results are returned in three columns. The first column is the network string ID. The second column is the string itself. The third column is the reference count to the network string.

**Note:**

This function is available only in debug builds.

```c
string getTag (string textTagString)
```

Extracts the tag from a tagged string.

Should only be used within the context of a function that receives a tagged string, and is not meant to be used outside of this context.

**Parameters:**

- `textTagString` The tagged string to extract.

**Returns:**

- The tag ID of the string.

See also:
**Data Types** under Tagged Strings

**detag()**

```cpp
string getTaggedString(int tag )
```

Use the `getTaggedString` function to convert a tag to a string.

This is not the same as `detag()` which can only be used within the context of a function that receives a tag. This function can be used any time and anywhere to convert a tag to a string.

**Parameters:**

- `tag` A numeric tag ID.

**Returns:**

The string as found in the Net String table.

**See also:**

- Data Types under Tagged Strings
- `addTaggedString()`
- `removeTaggedString()`

```cpp
void onDataBlockObjectReceived(int index, int total )
```

Called on the client each time a datablock has been received.

This callback is typically used to notify the player of how far along in the datablock download process they are.

**Parameters:**

- `index` The index of the datablock just received.
- `total` The total number of datablocks to be received.
See also:

GameConnection, GameConnection::transmitDataBlocks(), GameConnection::onDataBlocksDone()

```cpp
void onLightManagerActivate(string name )
```

A callback called by the engine when a light manager is activated.

**Parameters:**

- `name` The name of the light manager being activated.

```cpp
void onLightManagerDeactivate(string name )
```

A callback called by the engine when a light manager is deactivated.

**Parameters:**

- `name` The name of the light manager being deactivated.

```cpp
void pathOnMissionLoadDone( )
```

Load all Path information from the mission.

This function is usually called from the loadMissionStage2() server-side function after the mission file has loaded. Internally it places all Paths into the server's PathManager. From this point the Paths are ready for transmission to the clients.

**Example:**

```cpp
// Inform the engine to load all Path info
pathOnMissionLoadDone();
```

See also:
void removeTaggedString(int tag)

Remove a tagged string from the Net String Table.

**Parameters:**

*tag* The tag associated with the string

**See also:**

Data Types under Tagged Strings
addTaggedString()
getTaggedString()

bool setNetPort(int port,
    bool bind = true)

Set the network port for the game to use.

**Parameters:**

*port* The port to use.
*bind* True if bind() should be called on the port.

**Returns:**

True if the port was successfully opened. This will trigger a windows firewall prompt. If you don't have firewall tunneling tech you can set this to false to avoid the prompt.
### Variable Documentation

**void allowConnections**

Sets whether or not the global NetInterface allows connections from remote hosts.

```c
void allowConnections(bool allow);
```

**Parameters:**

- `allow` Set to true to allow remote connections.

**int $pref::Net::LagThreshold**

How long between received packets before the client is considered as lagging (in ms).

This is used by `GameConnection` to determine if the client is lagging. If the client is indeed lagging, `setLagIcon()` is called to inform the user in some way. i.e. display an icon on screen.

**See also:**

- `GameConnection`, `GameConnection::setLagIcon()`

**int $Stats::netBitsReceived**

The number of bytes received during the last packet process operation.

**Note:**

Even though this variable has 'Bits' in it, the value is indeed reported in bytes. This name is a legacy holdover and remains for compatibility reasons.
**int $Stats::netBitsSent**

The number of bytes sent during the last packet send operation.

**Note:**
Even though this variable has 'Bits' in it, the value is indeed reported in bytes. This name is a legacy holdover and remains for compatibility reasons.

**int $Stats::netGhostUpdates**

The total number of ghosts added, removed, and/or updated on the client during the last packet process operation.

**int $pref::Net::PacketRateToClient**

Sets how often packets are sent from the server to a client.

It is possible to control how often packets may be sent to the clients. This may be used to throttle the amount of bandwidth being used, but should be adjusted with caution.

The actual formula used to calculate the delay between sending packets to a client is:

\[
\text{Packet Update Delay To Client} = \frac{1024}{\$pref}
\]

with the result in ms. A minimum rate of 1 is enforced in the source code. The default value is 10.

**Note:**
When using a local connection ([Local Connections](#)) be aware that this variable is always forced to 128.
**int $pref::Net::PacketRateToServer**

Sets how often packets are sent from the client to the server.

It is possible to control how often packets may be sent to the server. This may be used to throttle the amount of bandwidth being used, but should be adjusted with caution.

The actual formula used to calculate the delay between sending packets to the server is:

\[
\text{Packet Update Delay To Server} = \frac{1024}{\$pref::Net::PacketRateToServer}
\]

with the result in ms. A minimum rate of 8 is enforced in the source code. The default value is 32.

**Note:**

When using a local connection ([Local Connections](#)) be aware that this variable is always forced to 128.

**int $pref::Net::PacketSize**

Sets the maximum size in bytes an individual network packet may be.

It is possible to control how large each individual network packet may be. Increasing its size from the default allows for more data to be sent on each network send. However, this value should be changed with caution as too large a value will cause packets to be split up by the networking platform or hardware, which is something Torque cannot handle.

A minimum packet size of 100 bytes is enforced in the source code. There is no enforced maximum. The default value is 200 bytes.

**Note:**
When using a local connection (Local Connections) be aware that this variable is always forced to 1024 bytes.

Copyright © GarageGames, LLC. All Rights Reserved.
Platform
## Classes

<table>
<thead>
<tr>
<th>class</th>
<th>RigidShape</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The RigidShape class implements rigid-body physics for DTS objects in the world. More...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th>RigidShapeData</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Defines the physics properties for an individual RigidShapeData physics object. More...</td>
</tr>
</tbody>
</table>
## Modules

<table>
<thead>
<tr>
<th>Localization</th>
<th>Classes and functions related Torque 3D networking.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mac</td>
<td>Functionality exclusive to Torque on OSX.</td>
</tr>
<tr>
<td>Windows</td>
<td>Functionality exclusive to Torque on Windows.</td>
</tr>
</tbody>
</table>
## Enumerations

**MBButtons**

```c
enum MBButtons {
    Ok,
    OkCancel,
    RetryCancel,
    SaveDontSave,
    SaveDontSaveCancel
}
```

Which buttons to display on a message box.

**More...**

**MBIcons**

```c
enum MBIcons {
    Information,
    Warning,
    Stop,
    Question
}
```

What icon to show on a message box.

**More...**

**MBReturnVal**

```c
enum MBReturnVal {
    OK,
    Cancelled,
    Retry,
    DontSave
}
```

Return value for `messageBox()` indicating which button was pressed by the user.

**More...**
**Functions**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bool displaySplashWindow()</code></td>
<td>Display a startup splash window suitable for showing while the engine still starts up.</td>
</tr>
<tr>
<td><code>int getRealTime()</code></td>
<td>Return the current real time in milliseconds.</td>
</tr>
<tr>
<td><code>int getSimTime()</code></td>
<td>Sim time is time since the game started.</td>
</tr>
<tr>
<td><code>bool getWebDeployment()</code></td>
<td>Test whether Torque is running in web-deployment mode.</td>
</tr>
<tr>
<td><code>void gotoWebPage (string address)</code></td>
<td>Open the given URL or file in the user's web browser.</td>
</tr>
<tr>
<td><code>bool isDebugBuild()</code></td>
<td>Test whether the engine has been compiled with TORQUE_DEBUG, i.e. if it includes debugging functionality.</td>
</tr>
<tr>
<td><code>bool isShippingBuild()</code></td>
<td>Test whether the engine has been compiled with TORQUE_SHIPPING, i.e. in a form meant for final release.</td>
</tr>
<tr>
<td><code>bool isToolBuild()</code></td>
<td>Test whether the engine has been compiled with TORQUE_TOOLS, i.e. if it includes tool-related functionality.</td>
</tr>
<tr>
<td><code>int messagebox (string title, string message, MBBUTTONS buttons=MBOkCancel, MBICONs icons=MIInformation)</code></td>
<td>Display a modal message box using the platform's native message box implementation.</td>
</tr>
<tr>
<td><code>void playJournal (string filename)</code></td>
<td>Begin playback of a journal from a specified field.</td>
</tr>
<tr>
<td><code>void quit ()</code></td>
<td>Shut down the engine and exit its process.</td>
</tr>
<tr>
<td><code>void quitWithErrorMessage (string message)</code></td>
<td>Shut down the engine and exit its process.</td>
</tr>
</tbody>
</table>
Display an error message box showing the given *message* and then shut down the engine and exit its process.

<table>
<thead>
<tr>
<th>void</th>
<th><code>saveJournal</code> (string <code>filename</code>)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Save the journal to the specified file.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th><code>shellExecute</code> (string <code>executable</code>, string <code>args</code>, string <code>directory</code>)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Launches an outside executable or batch file.</td>
</tr>
</tbody>
</table>
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>const int</td>
<td>$MRCancel</td>
<td>Determines the cancel button press state in a message box.</td>
</tr>
<tr>
<td>const int</td>
<td>$MRDontSave</td>
<td>Determines the don't save button press state in a message box.</td>
</tr>
<tr>
<td>const int</td>
<td>$MROk</td>
<td>Determines the ok button press state in a message box.</td>
</tr>
<tr>
<td>const int</td>
<td>$MRRetry</td>
<td>Determines the retry button press state in a message box.</td>
</tr>
<tr>
<td>int</td>
<td>$platform::backgroundSleepTime</td>
<td>Controls processor time usage when the game window is out of focus.</td>
</tr>
<tr>
<td>int</td>
<td>$platform::timeManagerProcessInterval</td>
<td>Controls processor time usage when the game window is in focus.</td>
</tr>
</tbody>
</table>
Detailed Description
**Enumeration Type Documentation**

**enum MBButtons**

Which buttons to display on a message box.

**Enumerator:**

- Ok
- OkCancel
- RetryCancel
- SaveDontSave
- SaveDontSaveCancel

**enum MBIcons**

What icon to show on a message box.

**Enumerator:**

- Information
- Warning
- Stop
- Question

**enum MBReturnVal**

Return value for `messageBox()` indicating which button was pressed by the user.

**Enumerator:**

- OK
- Cancelled
- Retry
- DontSave
Function Documentation

bool displaySplashWindow( )

Display a startup splash window suitable for showing while the engine still starts up.

Note:
This is currently only implemented on Windows.

Returns:
True if the splash window could be successfully initialized.

int getRealTime( )

Return the current real time in milliseconds.

Real time is platform defined; typically time since the computer booted.

int getSimTime( )

Sim time is time since the game started.

Return the current sim time in milliseconds.

bool getWebDeployment( )

Test whether Torque is running in web-deployment mode.

In this mode, Torque will usually run within a browser and certain restrictions apply (e.g. Torque will not be able to enter fullscreen exclusive mode).
Returns:
True if Torque is running in web-deployment mode.

```c
void gotoWebPage(string address)
```

Open the given URL or file in the user's web browser.

Parameters:
The address to open. If this is not prefixed by a protocol specifier ("...://"), then the function checks `address` whether the address refers to a file or directory and if so, prepends "file://" to `address`; if the file check fails, "http://" is prepended to `address`.

Example:
```c
gotoWebPage( "http://www.garagegames.com"
```

```c
bool isDebugBuild()
```

Test whether the engine has been compiled with TORQUE_DEBUG, i.e. if it includes debugging functionality.

Returns:
True if this is a debug build; false otherwise.

```c
bool isShippingBuild()
```

Test whether the engine has been compiled with TORQUE_SHIPPING, i.e. in a form meant for final release.

Returns:
True if this is a shipping build; false otherwise.
bool isToolBuild( )

Test whether the engine has been compiled with TORQUE_TOOLS, i.e. if it includes tool-related functionality.

**Returns:**
True if this is a tool build; false otherwise.

```c
int messagebox(string title, string message, MBButtons buttons = MBOkCancel, MBIcons icons = MIInformation )
```

Display a modal message box using the platform's native message box implementation.

**Parameters:**
- `title` The title to display on the message box window.
- `message` The text message to display in the box.
- `buttons` Which buttons to put on the message box.
- `icons` Which icon to show next to the message.

**Returns:**
One of $MROK, $MRCancel, $MRRetry, and $MRDontSave identifying the button that the user pressed.

**Example:**
```c
messageBox( "Error", "" );
```

```c
void playJournal(string filename )
```

Begin playback of a journal from a specified field.
**Parameters:**

*filename* Name and path of file journal file

```c
void quit()
```

Shut down the engine and exit its process.

This function cleanly uninitializes the engine and then exits back to the system with a process exit status indicating a clean exit.

**See also:**

* quitWithErrorMessage *

```c
void quitWithErrorMessage(string message)
```

Display an error message box showing the given *message* and then shut down the engine and exit its process.

This function cleanly uninitialized the engine and then exits back to the system with a process exit status indicating an error.

**Parameters:**

* message* The message to log to the console and show in an error message box.

**See also:**

* quit *

```c
void saveJournal(string filename)
```

Save the journal to the specified file.

```c
bool shellExecute(string executable,
```
Launches an outside executable or batch file.

**Parameters:**

- **executable**  Name of the executable or batch file
- **args**  Optional list of arguments, in string format, to pass to the executable
- **directory**  Optional string containing path to output or shell
### Variable Documentation

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int $MRCancel</td>
<td>Determines the cancel button press state in a message box.</td>
</tr>
<tr>
<td>int $MRDontSave</td>
<td>Determines the don't save button press state in a message box.</td>
</tr>
<tr>
<td>int $MROk</td>
<td>Determines the ok button press state in a message box.</td>
</tr>
<tr>
<td>int $MRRetry</td>
<td>Determines the retry button press state in a message box.</td>
</tr>
<tr>
<td>int $platform::backgroundSleepTime</td>
<td>Controls processor time usage when the game window is out of focus.</td>
</tr>
<tr>
<td>int $platform::timeManagerProcessInterval</td>
<td>Controls processor time usage when the game window is in focus.</td>
</tr>
</tbody>
</table>
Localization
[Platform]

Classes and functions related Torque 3D networking. More...
## Classes

<table>
<thead>
<tr>
<th>class</th>
<th>LangTable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Provides the code necessary to handle the low level management of the string tables for localization. <a href="#">More...</a></td>
</tr>
</tbody>
</table>
## Functions

<table>
<thead>
<tr>
<th>int getCoreLangTable ()</th>
<th>Gets the primary <code>LangTable</code> used by the game.</th>
</tr>
</thead>
<tbody>
<tr>
<td>void setCoreLangTable (string LangTable)</td>
<td>Sets the primary <code>LangTable</code> used by the game.</td>
</tr>
</tbody>
</table>
Detailed Description

Classes and functions related Torque 3D networking.

Overview
Introduction

Localizing large applications can be an extremely time consuming and aggravating process. This manual is intended to guide you through the changes made to Torque to support localization and help you reduce the ongoing headaches in day-to-day development in a localized codebase.

This manual assumes you are at least familiar with Torque Script. C++ and localization experience is not necessary.
Languages and String Tables

Lets say you want to print some text to the console. From script, you’d usually just write something like:

echo("Hello, World!");

This is all well and good, if all you care about is English. What do you do if the user wants your game to talk French? You could do something like the following:

switch($Pref::Language)
{
  case $ENGLISH:
    echo("Hello, World!");
  case $FRENCH:
    echo("Salut, Monde!");
... other languages here ...
}

Sure, this works. It’s also a complete pain in the backside to keep up to date, let alone that you’d have to repeat the switch for every string you use. In the C++ code alone there is upwards of 4000 strings that are candidates for localization, so this method is clearly not even close to an option.

String tables are the answer to this problem. A string table is an array of strings that you can reference based on an ID instead of specifying the string directly in the source code. Changing which language the string is displayed in is simply a matter of using a string table that contains the strings in the language you wish to display. The above example can be reduced down to the following:

echo(L($STR_HELLO_WORLD));

The code is not much different to how you wrote it before, but the string will now be displayed in the correct language based on the user’s selection. This does come with a price, however. You now have to create and maintain string tables for all the languages you support, which can be a big headache. Torque tries to make this as simple as possible, and some tips for creating and maintaining these
string tables are given later.
Script Interface

Torque contains a lot of strings, not just in the C++ code but in the script code too. For example, all your GUIs are created in script and must also be localized. To further complicate matters the scripts are contained in mod directories, each of which is a separate entity.

Strings in Script

It would be cumbersome and difficult to maintain one string table for all the strings, so Torque doesn't try. Each mod has it's own string table, as well as the core C++ code. You would be forgiven for thinking that this leads to having to write code as follows:

```plaintext
echo(L($StarterFPSTable, $STR_HELLO_WORLD));
```

I don’t know about you, but I'm a lazy programmer and that's just too much typing for me to put up with every time I want to display a string. Instead, you just use:

```plaintext
echo(L($STR_HELLO_WORLD));
```

Torque takes care of the rest: the correct table will be used based on which mod the code is executed from.

Loading strings and specifying languages

The new LangTable class, accessible in script as well as in C++, provides the code necessary to handle the low level management of the string tables and obtaining the correct strings from them. One LangTable is created for each mod, as well as one for the C++ code.

An important portion of the localization system is implemented in script. This support code is responsible for managing the language tables and provides you with a simpler API than would be the case using LangTable directly. To provide this API, a new mod called "lang" has been added. The lang mod works a little like the common mod in that it just provides code needed by all mods. As the localization code is needed very early in the initialisation code, adding it to the common mod was not possible.
To ease loading of language tables, language maps are provided. These map the displayed name of the language to a string table on disk and are used to provide automatic loading across all mods. You specify the language map in the lang/languages.cs script, for example:

```csharp
addLanguageMap("English", "english.lso");
addLanguageMap("Pony", "pony.lso");
```

The first argument to addLanguageMap() is the name of the language as it will be displayed to the user, and the second is the filename of the language. The filename will be concatenated onto the mod path to find the actual filename name of the table to load.

To load the language tables for the current mod, in the onStart() function for the mod, before any other code, use the following:

```csharp
table = loadModLangTables("starter.fps/lang/tables");
setModLangTable(table);
```

loadModLangTables() loads the tables from the specified directory based on the information in the language map. setModLangTable() tells the C++ code to use the specified table for the current mod. This is usually done in the mod's onStart() function. After loading the tables, the current language will be set to English.

```csharp
// setup localization:
$I18N::starter.fps = new LangTable();
exec("starter.fps/lang/tables/German.cs");
exec("starter.fps/lang/tables/English.cs");
$I18N::starter.fps.addLanguage("starter.fps/lang/tables/German.lso", "German");
$I18N::starter.fps.addLanguage("starter.fps/lang/tables/English.lso", "English");
$I18N::starter.fps.setDefaultLanguage(0); // German is default
$I18N::starter.fps.setCurrentLanguage(0); // German is current
$I18N::starter.fps.setCurrentLanguage(1); // this would set the cu
```

**Handling user language preferences**

In the Torque demo, the user is provided two ways to change their language settings. If they have not already chosen a language, for example the first time they run the game, a dialog will pop up after the splash screens asking them to choose a language. The language can also be changed at any time from the options dialog. Note that
some strings will not be in the new language until the game is restarted. There will be further discussion on this topic in later chapters.

When the language is changed, the `onChangeLanguage()` function is called. This should be overridden for each mod in the same way as `onStart()`. The following code can be used for all mods:

```plaintext
function onChangeLanguage()
{
    setCurrentLanguage(getModLangTable(), $pref::I18N::language);
    Parent::onChangeLanguage();
}
```

To change the language for all mods, you can simply use the `setLanguage()` function. `setLanguage()` takes one argument, the display name of the language to switch to.
Localizing the GUI

The GUI provides some interesting challenges for localization. Since GUI files are just script files that create the objects for the GUI, text is specified as a member field of the object. For example:

```java
new GuiButtonCtrl() {
    text = "Quit!";
    ... other fields ...
};
```

As this is just a script, you can localize it easily by using the L() function in the same way you would with other scripts, for example:

```java
new GuiButtonCtrl() {
    text = L($STR_QUIT);
    ... other fields ...
};
```

Unfortunately, this is not a viable solution. If you save that GUI from the GUI editor, the call to L() will be replaced with the text it returned when creating the GUI, reverting it to the first example. The only way around that is to edit all your GUIs by hand, which is not a particularly pleasant option when you have an editor to do it for you.

In order to sensibly support editing of GUIs, most of the GUI controls that support the text field now also support a textID field. This is the name of the ID variable for the string, for example:

```java
new GuiButtonCtrl() {
    textID = "STR_QUIT"
    ... other fields ...
};
```

In addition, a setTextID() method was added to all controls that expose a setText() method to allow you to set the text at run time. It functions the same as the setText() method, except it works with IDs.

```java
obj.setTextID("STR_ABOUT");
```

The only caveat is you must specify the name of the variable as a string, and not the variable itself. If you used the variable, for
example obj.setText($STR_ABOUT), then the textID field would get set to the numeric ID, and the GUI control would be looking for a variable called, for example, 45.

If you don't specify an ID, or the ID you do specify is invalid for whatever reason, then the control will simply use the text field as before.

**Specifying the language table**

Unfortunately, there is not enough context available in the GUI system to determine which mod the control was defined in, so the tricks used to make the L() function work cannot be used in the GUI.

To work around the problem, a field was added to `GuiControl` to specify the table to use, called langTableMod, which is simply the name of the mod to get the language table from. Usually, this will be the same as the mod that the `.gui` file resides in.

To avoid having to specify the table for every control that needs it, a crude form of inheritance was implemented. If langTableMod is not specified for a control, the control's parent will be checked. This means you only have to specify the table in the root control for the GUI. However, you could also specify a different table on a per-control basis if needed, for example if you wanted to use a string from the language table for common. The following example GUI illustrates this better:

```java
// The value that you set langTableMod to is the same as name a

new GuiChunkedBitmapCtrl(MainMenuGui) {
    profile = "GuiContentProfile";
    horizSizing = "width";
    vertSizing = "height";
    position = "0 0";
    extent = "640 480";
    minExtent = "8 8";
    visible = "1";
    langTableMod = "starter.fps";
    bitmap = "./background";
    useVariable = "0";
    tile = "0";
    helpTag = "0";
};
```
new GuiButtonCtrl() {
    profile = "GuiButtonProfile";
    horizSizing = "right";
    vertSizing = "top";
    position = "36 413";
    extent = "110 20";
    minExtent = "8 8";
    visible = "1";
    command = "quit();";
    textID = "STR_QUIT";
    groupNum = "-1";
    buttonType = "PushButton";
    helpTag = "0";
};

Refreshing the GUI

When the user has chosen a new language, it would be nice to be able to get the GUI to refresh without forcing them to restart the game. One nice side effect of the updates to the GUI system is that we can do that.

The way the textID field was implemented is it checks for a valid ID, then if it has one it passes the string directly to setText(). The initial update of the text field is done on onWake(), so to get the control to update you only have to cause it to get an onWake() event.

It turns out that you can trick Torque into resending onWake() events quite simply, without having to write a ton of code. For the main canvas, you can simply do:

    if(isObject(Canvas))
    Canvas.setContent(Canvas.getContent());

When implementing the options dialog, I found that this won't refresh the dialogs. To work around that, when you click apply the script will pop the dialog then re-push it, as follows:

    Canvas.popDialog(optionsDlg);
    Canvas.pushDialog(optionsDlg);

Of course, some things will not be possible to update when you change the language. It may be worth considering requiring users to
restart the game for language changes to take effect for the sake of consistency.
C++ Interface

The C++ interface is largely in flux at the moment pending fixing the string extractor to cope with multi-line strings. In terms of how you reference strings, it will be largely the same as the script interface.
Caveats

This will have caveats for the localization thingy.
The Localization Tools

The current toolset consists of the command line language compiler, called langc, which is used to compile a language file into the various formats needed by the engine.

Language Files

Language files are simple text files containing the identifiers and strings. An example follows:

```
TEST_STR_1=This is a test
TEST_STR_2=and so is this!
```

The string definitions are similar to variable definitions in C++ and Torque Script, except you do not need quotes or a semicolon. The "variable" becomes the ID as it will be referred to in the game code.

If you need the string to span multiple lines, you can use the usual as you would in C++. Note that currently you can't use a trailing \ on the line to continue to the next line. *(Note: this should be fixed)*

You can use any amount of white space on either side of the equals sign. Two options control how the compiler handles whitespace. By default, any whitespace after the equals (leading spaces) will be stripped and any space on the end of the string (trailing spaces) will be left alone. Two options control this behaviour:

-S

Strip leading space. When this option is specified, any space after the = sign will become part of the string.

-T

Strip trailing space. When specified, space on the end of the string will be stripped. </tbody>
Comments may be inserted in the language file at the beginning of a line only. You can use #, ; or // to delimit comments. For example:

# This is a comment
; So is this
And this

**Compiling Language Files**

Before a language file can be used in the engine, it has to be compiled into a .lso file. As the process is slightly different depending on whether you are compiling the default English language file or a translation, it is not possible to do this automatically, as is done with scripts.

The default language is usually English. There is no hard and fast restriction on which language is the default, but for the purposes of documentation it will be assumed that English is your default language. It is worth pointing out that because strings that are missing from a particular language will use the default language, and some strings (particularly in the C++ code) are not localized, not using English for the default language may cause inconsistencies.

Language files are compiled with the langc tool, found in the tools directory. Like other Torque tools, langc must be compiled before you can use it.

langc can also produce some other related files, for example header files for the IDs and templates for translations.

In its simplest usage, langc requires the name of the language file to compile and an output basename, for example:

> langc english.lang english

The first argument is the name of the language file and the second is the basename for output files. langc can create a number of different files, so rather then force you to specify the filename of each one explicitly, you only have to specify the first part of the filename. The relevant extension will be added automatically.
If you were to run the above example, `english.lang` would be compiled, but no output files will be generated. This can be useful for checking if there are any errors in the file, but most of the time you'll want to produce some output files.

To compile the language to an Iso, you use the `-l` option, as follows:

```
> langc -l english.lang english
```

This will compile `english.lang` and, assuming there were no errors, will produce `english.iso`, ready to load into the engine.

**Identifiers**

Earlier in this manual, global variables were used as identifiers for the strings, but there was no mention of where they came from. Because these are a pain in the backside to maintain, `langc` does the job for you. When you compile the English language file, you can also generate a script file containing global variables for each string ID in the language file.

To generate the script, simply use the `-s` option. For example:

```
> langc -s english.lang english
```

This will generate a file called `english.cs`. Using the earlier example `.lang` file, the generated script would look something like this:

```latex
// Automatically generated. DO NOT EDIT
This is a test
$TEST_STR_1 = 0;
and so is this!
$TEST_STR_2 = 1;
```

If you'd like to save some time, you can specify multiple options at once, as in the following example:

```
> langc -ls english.lang english
```

This will create both `english.iso` and `english.cs`. 
Compiling Translations

Although translations compile to the same .lso files as the default language, the process is slightly more involved. This is because the identifiers must come from the default English file to ensure they are correct.

You can create a template for a new translation, based on the English file, as follows:

```
> langc -r english.lang french
```

This will generate a file called french.tran, the format of which is identical to the .lang files described earlier.

```
# Automatically generated.
# [TEST_STR_1:0] This is a test
TEST_STR_1=
# [TEST_STR_2:1] and so is this!
TEST_STR_2=
```

The comments preceding each string definition are intended to tell the translator what the original English string was. The actual string definitions are left blank so that langc will issue a warning for empty, and thus untranslated, strings.

Compiling a translation requires a few more options then compiling the English file.

```
> langc -t -l english.lang french.tran french
```

Here, -t tells langc you want to compile a translation, -e English.lang specifies the filename of the English translation and, as before, -l causes the .lso file to be compiled.

When the translation is loaded into the engine, any strings that are left empty will automatically fall back to the English version of them. By default, langc will issue a warning when compiling the file if a string is empty. This is useful while compiling translations to determine which strings have not been translated. However, it's less
useful when compiling the english file, so these warnings may be
turned off with the -W option. It is not recommended to use -W when
compiling translations.
Other langc options

Aside from the above, langc provides a few additional options that you may find useful. These were purposefully not mentioned before as they are not particularly useful at this time.

A complete list of current langc options may be obtained by running langc with no arguments, as follows:

> langc
Usage: langc [options] <filename> <outbasename>

Where options is one or more of:

-\*l\* Write Language File
-\*h\* Write C++ Header
-\*s\* Write Script
-\*d\* Write C++ Defaults
-\*t\* Compile a translation
-\*r\* Write translation fi
-\*e\* <filename> Specify english file when compiling translations
-\*S\* Don't strip leading spaces
-\*T\* Strip trailing space
-\*I\* Don't warn for invalid chars
-\*W\* Don't warn for empty
-\*q\* Quiet mode, no warnings at all

Of particular note are the options -h and -d, used when building files for the C++ localization. The -h option generates a C++ header file in the same way as the -s option does for scripts. The -d option writes a C++ source file that provides a big array of default strings. This is intended to be used as an extra fallback when a string can’t be found in the language table (or when there is no language table), which allows the same executable to be used whether localization is active or not.

Identifiers must be valid variable names, as they are used directly both as script variables and C++ defines. By default, langc will warn you if an invalid character is used in an identifier. The -l option prevents this.

The -q option is simply a shortcut to disable all warnings at once. If any errors occur, they will still be displayed.
Function Documentation

```c
int getCoreLangTable( )
```

Gets the primary `LangTable` used by the game.

**Returns:**
- ID of the core `LangTable`

```c
void setCoreLangTable( string LangTable )
```

Sets the primary `LangTable` used by the game.

**Parameters:**
- `LangTable` ID of the core `LangTable`

Copyright © GarageGames, LLC. All Rights Reserved.
Functionality exclusive to Torque on OSX. More...

Functionality exclusive to Torque on OSX.

Copyright © GarageGames, LLC. All Rights Reserved.
Windows
[Platform]

Functionality exclusive to Torque on Windows. More...

Functionality exclusive to Torque on Windows.

Copyright © GarageGames, LLC. All Rights Reserved.
Rendering

All rendering related functionality. More...
## Classes

<table>
<thead>
<tr>
<th>class</th>
<th>PostEffect</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A fullscreen shader effect. <a href="#">More...</a></td>
</tr>
</tbody>
</table>
## Modules

<table>
<thead>
<tr>
<th>Font</th>
<th>Various helpers for working with fonts from script.</th>
</tr>
</thead>
<tbody>
<tr>
<td>GFX</td>
<td>The low level graphics interface to the engine.</td>
</tr>
<tr>
<td>Lighting</td>
<td>The script functionality related to the lighting systems and lights.</td>
</tr>
<tr>
<td>Render Binning</td>
<td>The render sorting and batching system.</td>
</tr>
</tbody>
</table>
### Enumerations

<table>
<thead>
<tr>
<th>Enum</th>
<th>PFXRenderTime {</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PFXBeforeBin,</td>
</tr>
<tr>
<td></td>
<td>PFXAfterBin,</td>
</tr>
<tr>
<td></td>
<td>PFXAfterDiffuse,</td>
</tr>
<tr>
<td></td>
<td>PFXEndOfFrame,</td>
</tr>
<tr>
<td></td>
<td>PFXTexGenOnDemand</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

When to process this effect during the frame.

**More...**

<table>
<thead>
<tr>
<th>Enum</th>
<th>PFXTargetClear {</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PFXTargetClear_None,</td>
</tr>
<tr>
<td></td>
<td>PFXTargetClear_OnCreate,</td>
</tr>
<tr>
<td></td>
<td>PFXTargetClear_OnDraw</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

Describes when the target texture should be cleared.

**More...**
### Functions

**void addGlobalShaderMacro (string name, string value=NULL)**

Adds a global shader macro which will be merged with the script defined macros on every shader. The macro will replace the value of an existing macro of the same name. For the new macro to take effect all the shaders in the system need to be reloaded.

**void beginSampling (location,[backend])**

Takes a string informing the backend where to store sample data and optionally a name of the specific logging backend to use. The default is the CSV backend. In most cases, the logging store will be a file name.

**void enableSamples (pattern,[state])**

Enable sampling for all keys that match the given name pattern. Slashes are treated as separators.

**int getActiveDDSFiles ()**

Returns the count of active DDSs files in memory.

**String getBitmapInfo (string filename)**

Returns image info in the following format: width TAB height TAB bytesPerPixel. It will return an empty string if the file is not found.

**void initDisplayDeviceInfo ()**

Initializes variables that track device and vendor information/IDs.

**void playJournalToVideo (string journalFile, string videoFile=NULL, string encoder="THEORA", float framerate=30.0f, Point2I resolution=Point2I(0, 0))**

Load a journal file and capture it video.

**void removeGlobalShaderMacro (string name)**

Removes an existing global macro by name.

**void startVideoCapture (GuiCanvas canvas, string filename, string encoder="THEORA", float framerate=30.0f, Point2I**
<table>
<thead>
<tr>
<th>reset</th>
<th>resolution=Point2I(0, 0))</th>
<th>Begins a video capture session.</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>stopSampling ()</td>
<td>Stops the rendering sampler.</td>
</tr>
<tr>
<td>void</td>
<td>stopVideoCapture ()</td>
<td>Stops the video capture session.</td>
</tr>
</tbody>
</table>
### Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>$\text{pref::imposter::canShadow}$</td>
<td>User preference which toggles shadows from imposters. Defaults to true.</td>
</tr>
<tr>
<td>float</td>
<td>$\text{pref::TS::detailAdjust}$</td>
<td>User preference for scaling the TSShape level of detail. The smaller the value the closer the camera must get to see the highest LOD. This setting can have a huge impact on performance in mesh heavy scenes. The default value is 1.</td>
</tr>
<tr>
<td>bool</td>
<td>$\text{Scene::disableTerrainOcclusion}$</td>
<td>Used to disable the somewhat expensive terrain occlusion testing.</td>
</tr>
<tr>
<td>bool</td>
<td>$\text{Scene::disableZoneCulling}$</td>
<td>If true, zone culling will be disabled and the scene contents will only be culled against the root frustum.</td>
</tr>
<tr>
<td>int</td>
<td>$\text{TSCore::frameCount}$</td>
<td>The number of frames that have been rendered since this control was created.</td>
</tr>
<tr>
<td>int</td>
<td>$\text{pref::Reflect::frameLimitMS}$</td>
<td>ReflectionManager tries not to spend more than this amount of time updating reflections per frame.</td>
</tr>
<tr>
<td>int</td>
<td>$\text{Sampler::frequency}$</td>
<td>Samples taken every nth frame.</td>
</tr>
<tr>
<td>bool</td>
<td>$\text{Scene::lockCull}$</td>
<td>Debug tool which locks the frustum culling to the current camera location.</td>
</tr>
<tr>
<td>int</td>
<td>$\text{pref::TS::maxInstancingVerts}$</td>
<td>Enables mesh instancing on non-skin meshes that have less that this count of verts. The default value is 200. Higher values can degrade performance.</td>
</tr>
<tr>
<td>int</td>
<td>$\text{Scene::maxOccludersPerZone}$</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Variable</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Float</td>
<td>$\text{Scene}::\text{o}c\text{cluderMinHe}\text{ightPercentage}$</td>
<td>Maximum number of occluders that will be concurrently allowed into the scene culling state of any given zone.</td>
</tr>
<tr>
<td>Float</td>
<td>$\text{Scene}::\text{o}c\text{cluderMinW}\text{idthPercentage}$</td>
<td>TODO.</td>
</tr>
<tr>
<td>Float</td>
<td>$\text{pref}::\text{R}e\text{flect}::\text{r}e\text{fractTexScale}$</td>
<td>RefractTex has dimensions equal to the active render target scaled in both x and y by this float.</td>
</tr>
<tr>
<td>Bool</td>
<td>$\text{Scene}::\text{renderB}\text{oundingB}\text{oxes}$</td>
<td>If true, the bounding boxes of objects will be displayed.</td>
</tr>
<tr>
<td>Int</td>
<td>$\text{pref}::\text{TS}::\text{skipL}\text{oadDL}\text{s}$</td>
<td>User preference which causes TSShapes to skip loading higher lods. This potentialy reduces the GPU resources and materials generated as well as limits the LODs rendered. The default value is 0.</td>
</tr>
<tr>
<td>Int</td>
<td>$\text{pref}::\text{TS}::\text{skipR}\text{enderDL}\text{s}$</td>
<td>User preference which causes TSShapes to skip rendering higher lods. This will reduce the number of draw calls and triangles rendered and improve rendering performance when proper LODs have been created for your models. The default value is 0.</td>
</tr>
<tr>
<td>Float</td>
<td>$\text{pref}::\text{TS}::\text{smallestV}\text{i}b\text{iblePixelSize}$</td>
<td>User preference which sets the smallest pixel size at which TSShapes will skip rendering. This will force all shapes to stop rendering when they get smaller than this size. The default value is -1 which disables it.</td>
</tr>
<tr>
<td>Float</td>
<td>$\text{pref}::\text{windEffectR}\text{adius}$</td>
<td>Radius to affect the wind.</td>
</tr>
</tbody>
</table>
Detailed Description

All rendering related functionality.
### Enumeration Type Documentation

**enum PFXRenderTime**

When to process this effect during the frame.

**Enumerator:**

- **PFXBeforeBin**
  
  Before a RenderInstManager bin.

- **PFXAfterBin**
  
  After a RenderInstManager bin.

- **PFXAfterDiffuse**
  
  After the diffuse rendering pass.

- **PFXEndOfFrame**
  
  When the end of the frame is reached.

- **PFXTexGenOnDemand**
  
  This PostEffect is not processed by the manager. It will generate its texture when it is requested.

**enum PFXTargetClear**

Describes when the target texture should be cleared.

**Enumerator:**

- **PFXTargetClear_None**
  
  Never clear the PostEffect target.

- **PFXTargetClear_OnCreate**
  
  Clear once on create.
PFXTargetClear_OnDraw  Clear before every draw.
void addGlobalShaderMacro (string name,
            string value = NULL )

Adds a global shader macro which will be merged with the script defined macros on every shader. The macro will replace the value of an existing macro of the same name. For the new macro to take effect all the shaders in the system need to be reloaded.

See also:
        resetLightManager, removeGlobalShaderMacro

void beginSampling (location  )

Takes a string informing the backend where to store sample data and optionally a name of the specific logging backend to use. The default is the CSV backend. In most cases, the logging store will be a file name.

Example:
        beginSampling("mysamples.csv");

void enableSamples (pattern  )

Enable sampling for all keys that match the given name pattern. Slashes are treated as separators.

int getActiveDDSFiles (  )

Returns the count of active DDSs files in memory.
String getBitmapInfo(string filename)

Returns image info in the following format: width TAB height TAB bytesPerPixel. It will return an empty string if the file is not found.

void initDisplayDeviceInfo()

Initializes variables that track device and vendor information/IDs.

void playJournalToVideo(string journalFile,
                        string videoFile = NULL,
                        string encoder = "THEORA",
                        float framerate = 30.0f,
                        Point2I resolution = Point2I(0, 0))

Load a journal file and capture it video.

void removeGlobalShaderMacro(string name)

Removes an existing global macro by name.

See also:
    addGlobalShaderMacro

void startVideoCapture(GuiCanvas canvas,
                       string filename,
                       string encoder = "THEORA",
                       float framerate = 30.0f,
                       Point2I resolution = Point2I(0, 0))
Begins a video capture session.

**See also:**

stopVideoCapture

```java
void stopSampling()
```

Stops the rendering sampler.

```java
void stopVideoCapture()
```

Stops the video capture session.

**See also:**

startVideoCapture
## Variable Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>$pref::imposter::canShadow</td>
<td>User preference which toggles shadows from imposters. Defaults to true.</td>
</tr>
<tr>
<td>float</td>
<td>$pref::TS::detailAdjust</td>
<td>User preference for scaling the TSShape level of detail. The smaller the value the closer the camera must get to see the highest LOD. This setting can have a huge impact on performance in mesh heavy scenes. The default value is 1.</td>
</tr>
<tr>
<td>bool</td>
<td>$Scene::disableTerrainOcclusion</td>
<td>Used to disable the somewhat expensive terrain occlusion testing.</td>
</tr>
<tr>
<td>bool</td>
<td>$Scene::disableZoneCulling</td>
<td>If true, zone culling will be disabled and the scene contents will only be culled against the root frustum.</td>
</tr>
<tr>
<td>int</td>
<td>$TSControl::frameCount</td>
<td>The number of frames that have been rendered since this control was created.</td>
</tr>
<tr>
<td>int</td>
<td>$pref::Reflect::frameLimitMS</td>
<td></td>
</tr>
</tbody>
</table>
ReflectionManager tries not to spend more than this amount of time updating reflections per frame.

\[
\text{int } \$\text{Sampler}::\text{frequency}
\]

Samples taken every nth frame.

\[
\text{bool } \$\text{Scene}::\text{lockCull}
\]

Debug tool which locks the frustum culling to the current camera location.

\[
\text{int } \$\text{pref}::\text{TS}::\text{maxInstancingVerts}
\]

Enables mesh instancing on non-skin meshes that have less than this count of verts. The default value is 200. Higher values can degrade performance.

\[
\text{int } \$\text{Scene}::\text{maxOccludersPerZone}
\]

Maximum number of occluders that will be concurrently allowed into the scene culling state of any given zone.

\[
\text{float } \$\text{Scene}::\text{occluderMinHeightPercentage}
\]

TODO.

\[
\text{float } \$\text{Scene}::\text{occluderMinWidthPercentage}
\]
TODO.

float $pref::Reflect::refractTexScale

RefractTex has dimensions equal to the active render target scaled in both x and y by this float.

bool $Scene::renderBoundingBoxes

If true, the bounding boxes of objects will be displayed.

int $pref::TS::skipLoadDLs

User perference which causes TSShapes to skip loading higher lods. This potentially reduces the GPU resources and materials generated as well as limits the LODs rendered. The default value is 0.

See also:  
$pref::TS::skipRenderDLs

int $pref::TS::skipRenderDLs

User perference which causes TSShapes to skip rendering higher lods. This will reduce the number of draw calls and triangles rendered and improve rendering performance when proper LODs have been created for your models. The default value is 0.

See also:  
$pref::TS::skipLoadDLs
### float $pref::TS::smallestVisiblePixelSize$

User preference which sets the smallest pixel size at which TSShapes will skip rendering. This will force all shapes to stop rendering when they get smaller than this size. The default value is -1 which disables it.

### float $pref::windEffectRadius$

Radius to affect the wind.
Font
[Rendering]

Various helpers for working with fonts from script. More...
### Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void <code>dumpFontCacheStatus</code> ()</td>
<td>Dumps to the console a full description of all cached fonts, along with info on the codepoints each contains.</td>
</tr>
<tr>
<td>void <code>duplicateCachedFont</code> (string oldFontName, int oldFontSize, string newFontName)</td>
<td>Copy the specified old font to a new name. The new copy will not have a platform font backing it, and so will never have characters added to it. But this is useful for making copies of fonts to add postprocessing effects to via exportCachedFont.</td>
</tr>
<tr>
<td>void <code>exportCachedFont</code> (string faceName, int fontSize, string fileName, int padding, int kerning)</td>
<td>Export specified font to the specified filename as a PNG. The image can then be processed in Photoshop or another tool and reimported using importCachedFont. Characters in the font are exported as one long strip.</td>
</tr>
<tr>
<td>void <code>importCachedFont</code> (string faceName, int fontSize, string fileName, int padding, int kerning)</td>
<td>Import an image strip from exportCachedFont. Call with the same parameters you called exportCachedFont.</td>
</tr>
<tr>
<td>void <code>populateAllFontCacheRange</code> (int rangeStart, int rangeEnd)</td>
<td>Populate the font cache for all fonts with Unicode code points in the specified range.</td>
</tr>
<tr>
<td>void <code>populateAllFontCacheString</code> (string string)</td>
<td>Populate the font cache for all fonts with characters from the specified string.</td>
</tr>
<tr>
<td>void <code>populateFontCacheRange</code> (string faceName, int fontSize, int rangeStart, int rangeEnd)</td>
<td>Populate the font cache for the specified font with Unicode code points in the specified range.</td>
</tr>
<tr>
<td>void <code>populateFontCacheString</code> (string faceName, int fontSize, string string)</td>
<td>Populate the font cache for the specified font with characters</td>
</tr>
</tbody>
</table>
from the specified string.

<table>
<thead>
<tr>
<th>void writeFontCache ()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Force all cached fonts to serialize themselves to the cache.</td>
</tr>
</tbody>
</table>
Detailed Description

Various helpers for working with fonts from script.
Function Documentation

```c
void dumpFontCacheStatus( )
```

Dumps to the console a full description of all cached fonts, along with info on the codepoints each contains.

```c
void duplicateCachedFont( string oldFontName,
                          int oldFontSize,
                          string newFontName
)
```

Copy the specified old font to a new name. The new copy will not have a platform font backing it, and so will never have characters added to it. But this is useful for making copies of fonts to add postprocessing effects to via exportCachedFont.

**Parameters:**
- `oldFontName` The name of the font face to copy.
- `oldFontSize` The size of the font to copy.
- `newFontName` The name of the new font face.

```c
void exportCachedFont( string faceName,
                       int fontSize,
                       string fileName,
                       int padding,
                       int kerning
)
```

Export specified font to the specified filename as a PNG. The image can then be processed in Photoshop or another tool and reimported using importCachedFont. Characters in the font are exported as one long strip.
Parameters:

- **faceName** The name of the font face.
- **fontSize** The size of the font in pixels.
- **fileName** The file name and path for the output PNG.
- **padding** The padding between characters.
- **kerning** The kerning between characters.

```csharp
void importCachedFont(string faceName,
                      int fontSize,
                      string fileName,
                      int padding,
                      int kerning)
```

Import an image strip from exportCachedFont. Call with the same parameters you called exportCachedFont.

Parameters:

- **faceName** The name of the font face.
- **fontSize** The size of the font in pixels.
- **fileName** The file name and path for the input PNG.
- **padding** The padding between characters.
- **kerning** The kerning between characters.

```csharp
void populateAllFontCacheRange(int rangeStart,
                                int rangeEnd)
```

Populate the font cache for all fonts with Unicode code points in the specified range.

Parameters:

- **rangeStart** The start Unicode point.
- **rangeEnd** The end Unicode point.
**Note:**

We only support BMP-0, so code points range from 0 to 65535.

```csharp
void populateAllFontCacheString(string string )
```

Populate the font cache for all fonts with characters from the specified string.

```csharp
void populateFontCacheRange (string faceName, int fontSize, int rangeStart, int rangeEnd )
```

Populate the font cache for the specified font with Unicode code points in the specified range.

**Parameters:**

- `faceName`  The name of the font face.
- `fontSize`  The size of the font in pixels.
- `rangeStart`  The start Unicode point.
- `rangeEnd`  The end Unicode point.

**Note:**

We only support BMP-0, so code points range from 0 to 65535.

```csharp
void populateFontCacheString(string faceName, int fontSize, string string )
```
Populate the font cache for the specified font with characters from the specified string.

**Parameters:**

- `faceName` The name of the font face.
- `fontSize` The size of the font in pixels.
- `string` The string to populate.

```java
void writeFontCache()
```

Force all cached fonts to serialize themselves to the cache.

Copyright © GarageGames, LLC. All Rights Reserved.
GFX
[Rendering]

The low level graphics interface to the engine. More...
### Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CubemapData</strong></td>
<td>Used to create static or dynamic cubemaps.</td>
</tr>
<tr>
<td><strong>DebugDrawer</strong></td>
<td>A debug helper for rendering debug primitives to the scene.</td>
</tr>
<tr>
<td><strong>GFXCardProfiler</strong></td>
<td>Provides a device independent wrapper around both the capabilities reported by the card/drivers and the exceptions recorded in various scripts.</td>
</tr>
<tr>
<td><strong>GFXCardProfilerAPI</strong></td>
<td>This class is the interface between TorqueScript and GFXCardProfiler.</td>
</tr>
<tr>
<td><strong>GFXInit</strong></td>
<td>Functions for tracking GFX adapters and initializing them into devices.</td>
</tr>
<tr>
<td><strong>GFXSamplerStateData</strong></td>
<td>A sampler state used by GFXStateBlockData.</td>
</tr>
<tr>
<td><strong>GFXStateBlockData</strong></td>
<td>A state block description for rendering.</td>
</tr>
<tr>
<td><strong>Material</strong></td>
<td>A material in Torque 3D is a data structure that describes a surface.</td>
</tr>
<tr>
<td><strong>PfxVis</strong></td>
<td>Singleton class that exposes ConsoleStaticFunctions for debug visualizing PostEffects.</td>
</tr>
<tr>
<td><strong>RenderFormatToken</strong></td>
<td>Used to change the render target format when rendering in AL.</td>
</tr>
</tbody>
</table>
## Modules

<table>
<thead>
<tr>
<th>Materials</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classes, structures, functions, and variables related to Torque 3D's material system.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Shaders</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classes, structures, functions, and variables related to Torque 3D's shader system.</td>
</tr>
</tbody>
</table>
### Enumerations

<table>
<thead>
<tr>
<th>Enum</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>GFXAdapterType</code></td>
<td>Back-end graphics API used by the GFX subsystem.</td>
</tr>
<tr>
<td><code>GFXBlend</code></td>
<td>The supported blend modes.</td>
</tr>
<tr>
<td><code>GFXBlendOp</code></td>
<td></td>
</tr>
</tbody>
</table>

```c
enum GFXAdapterType {  
    OpenGL,  
    D3D8,   
    D3D9,   
    NullDevice,  
    Xenon  
}  

Back-end graphics API used by the GFX subsystem.

More...

enum GFXBlend {  
    GFXBlendZero,  
    GFXBlendOne,   
    GFXBlendSrcColor,  
    GFXBlendInvSrcColor,  
    GFXBlendSrcAlpha,  
    GFXBlendInvSrcAlpha,  
    GFXBlendDestAlpha,  
    GFXBlendInvDestAlpha,  
    GFXBlendDestColor,  
    GFXBlendInvDestColor,  
    GFXBlendSrcAlphaSat  
}  

The supported blend modes.

More...

enum GFXBlendOp {  
    GFXBlendOpAdd,  
    GFXBlendOpSubtract,  
    GFXBlendOpRevSubtract,  
    GFXBlendOpMin,  
    GFXBlendOpMax  
}  
```
The blend operators.

```cpp
enum GFXCmpFunc {
    GFXCmpNever,
    GFXCmpLess,
    GFXCmpEqual,
    GFXCmpLessEqual,
    GFXCmpGreater,
    GFXCmpNotEqual,
    GFXCmpGreaterEqual,
    GFXCmpAlways
}
```

The supported comparison functions.

```cpp
enum GFXCullMode {
    GFXCullNone,
    GFXCullCW,
    GFXCullCCW
}
```

The render cull modes.

```cpp
enum GFXFormat {
    GFXFormatR8G8B8,
    GFXFormatR8G8B8A8,
    GFXFormatR8G8B8X8,
    GFXFormatR32F,
    GFXFormatR5G6B5,
    GFXFormatR5G5B5A1,
    GFXFormatR5G5B5X1,
    GFXFormatA4L4,
    GFXFormatA8L8,
    GFXFormatA8,
    GFXFormatL8,
    GFXFormatR5G5B5A1,
    GFXFormatR5G5B5X1,
    GFXFormatA4L4,
    GFXFormatA8L8,
    GFXFormatA8,
    GFXFormatL8,
    GFXFormatR5G5B5A1,
    GFXFormatR5G5B5X1,
    GFXFormatA4L4,
    GFXFormatA8L8,
    GFXFormatA8,
    GFXFormatL8,
    GFXFormatR5G5B5A1,
    GFXFormatR5G5B5X1,
    GFXFormatA4L4,
    GFXFormatA8L8,
    GFXFormatA8,
    GFXFormatL8,
    GFXFormatR5G5B5A1,
    GFXFormatR5G5B5X1,
    GFXFormatA4L4,
    GFXFormatA8L8,
    GFXFormatA8,
    GFXFormatL8,
    GFXFormatR5G5B5A1,
    GFXFormatR5G5B5X1,
    GFXFormatA4L4,
    GFXFormatA8L8,
    GFXFormatA8,
    GFXFormatL8,
    GFXFormatR5G5B5A1,
    GFXFormatR5G5B5X1,
    GFXFormatA4L4,
    GFXFormatA8L8,
    GFXFormatA8,
    GFXFormatL8,
    GFXFormatR5G5B5A1,
    GFXFormatR5G5B5X1,
    GFXFormatA4L4,
    GFXFormatA8L8,
    GFXFormatA8,
    GFXFormatL8,
    GFXFormatR5G5B5A1,
    GFXFormatR5G5B5X1,
    GFXFormatA4L4,
    GFXFormatA8L8,
    GFXFormatA8,
    GFXFormatL8,
    GFXFormatR5G5B5A1,
    GFXFormatR5G5B5X1,
    GFXFormatA4L4,
    GFXFormatA8L8,
    GFXFormatA8,
    GFXFormatL8,
    GFXFormatR5G5B5A1,
    GFXFormatR5G5B5X1,
    GFXFormatA4L4,
    GFXFormatA8L8,
    GFXFormatA8,
    GFXFormatL8,
    GFXFormatR5G5B5A1,
    GFXFormatR5G5B5X1,
    GFXFormatA4L4,
    GFXFormatA8L8,
    GFXFormatA8,
    GFXFormatL8,
    GFXFormatR5G5B5A1,
    GFXFormatR5G5B5X1,
    GFXFormatA4L4,
    GFXFormatA8L8,
enum
GFXFormatDXT1,
GFXFormatDXT2,
GFXFormatDXT3,
GFXFormatDXT4,
GFXFormatDXT5,
GFXFormatD32,
GFXFormatD24X8,
GFXFormatD24S8,
GFXFormatD24FS8,
GFXFormatD16,
GFXFormatR32G32B32A32F,
GFXFormatR16G16B16A16F,
GFXFormatL16,
GFXFormatR16G16B16A16,
GFXFormatR16G16,
GFXFormatR16F,
GFXFormatR16G16F,
GFXFormatR10G10B10A2
}

The texture formats.

More...

enum
GFXStencilOp {
GFXStencilOpKeep,
GFXStencilOpZero,
GFXStencilOpReplace,
GFXStencilOpIncrSat,
GFXStencilOpDecrSat,
GFXStencilOpInvert,
GFXStencilOpIncr,
GFXStencilOpDecr
}

The stencil operators.

More...
<table>
<thead>
<tr>
<th>Enum Name</th>
<th>Enum Definition</th>
<th>Description</th>
<th>More...</th>
</tr>
</thead>
<tbody>
<tr>
<td>GFXTextureAddressMode</td>
<td>GFXTextureAddressMode { GFXAddressWrap, GFXAddressMirror, GFXAddressClamp, GFXAddressBorder, GFXAddressMirrorOnce }</td>
<td>The texture address modes.</td>
<td></td>
</tr>
<tr>
<td>GFXTextureArgument</td>
<td>GFXTextureArgument { GFXTADiffuse, GFXTACurrent, GFXTATexture, GFXTATFactor, GFXTASpecular, GFXTATemp, GFXTAClamp, GFXTAClamp, GFXTAClampNone, AlphaReplicate }</td>
<td>The texture arguments.</td>
<td></td>
</tr>
<tr>
<td>GFXTextureFilterType</td>
<td>GFXTextureFilterType { GFXTextureFilterNone, GFXTextureFilterPoint, GFXTextureFilterLinear, GFXTextureFilterAnisotropic, GFXTextureFilterPyramidalQuad, GFXTextureFilterGaussianQuad }</td>
<td>The texture filter types.</td>
<td></td>
</tr>
</tbody>
</table>
enum GFXTextureOp {
    GFXTOPDisable,
    GFXTOPSelectARG1,
    GFXTOPSelectARG2,
    GFXTOPModulate,
    GFXTOPModulate2X,
    GFXTOPModulate4X,
    GFXTOPAdd,
    GFXTOPAddSigned,
    GFXTOPAddSigned2X,
    GFXTOPSubtract,
    GFXTOPAddSmooth,
    GFXTOPBlendDiffuseAlpha,
    GFXTOPBlendTextureAlpha,
    GFXTOPBlendFactorAlpha,
    GFXTOPBlendTextureAlphaPM,
    GFXTOPBlendCURRENTALPHA,
    GFXTOPPreModulate,
    GFXTOPModulateAlphaAddColor,
    GFXTOPModulateColorAddAlpha,
    GFXTOPModulateInvAlphaAddColor,
    GFXTOPModulateInvColorAddAlpha,
    GFXTOPBumpEnvMap,
    GFXTOPBumpEnvMapLuminance,
    GFXTOPDotProduct3,
    GFXTOPLERP
}

The texture operators.

More...

enum GFXTextureTransformFlags {
    GFXTTFDisable,
    GFXTTFFCoord1D,
    GFXTTFFCoord2D,
    GFXTTFFCoord3D,
    GFXTTFFCoord4D,
    GFXTTFProjected
}
The texture transform state flags.

More...

<table>
<thead>
<tr>
<th>MaterialAnimType</th>
<th>{ Scroll, Rotate, Wave, Scale, Sequence }</th>
</tr>
</thead>
</table>

The type of animation effect to apply to this material.

More...

<table>
<thead>
<tr>
<th>MaterialBlendOp</th>
<th>{ None, Mul, Add, AddAlpha, Sub, LerpAlpha }</th>
</tr>
</thead>
</table>

The type of graphical blending operation to apply to this material.

More...

<table>
<thead>
<tr>
<th>MaterialWaveType</th>
<th>{ Sin, Triangle, Square }</th>
</tr>
</thead>
</table>

When using the Wave material animation, one of these Wave Types will be used to determine the type of wave to
display.

More...
## Functions

<table>
<thead>
<tr>
<th>void cleanupTexturePool ()</th>
<th>Release the unused pooled textures in texture manager freeing up video memory.</th>
</tr>
</thead>
<tbody>
<tr>
<td>void clearGFXResourceFlags ()</td>
<td>Clears the flagged state on all allocated GFX resources. See flagCurrentGFXResources for usage details.</td>
</tr>
<tr>
<td>void describeGFXResources (string resourceTypes, string filePath, bool unflaggedOnly=false)</td>
<td>Dumps a description of GFX resources to a file or the console.</td>
</tr>
<tr>
<td>void describeGFXStateBlocks (string filePath)</td>
<td>Dumps a description of all state blocks.</td>
</tr>
<tr>
<td>void dumpRandomNormalMap ()</td>
<td>Creates a 64x64 normal map texture filled with noise. The texture is saved to randNormTex.png in the location of the game executable.</td>
</tr>
<tr>
<td>void dumpTextureObjects ()</td>
<td>Dumps a list of all active texture objects to the console.</td>
</tr>
<tr>
<td>void flagCurrentGFXResources ()</td>
<td>Flags all currently allocated GFX resources. Used for resource allocation and leak tracking by flagging current resources then dumping a list of unflagged resources at some later point in execution.</td>
</tr>
<tr>
<td>void flushTextureCache ()</td>
<td>Releases all textures and resurrects the texture manager.</td>
</tr>
<tr>
<td>static int GFXInit::getAdapterCount ()</td>
<td>Return the number of graphics adapters available.</td>
</tr>
<tr>
<td>GFXFormat getBestHDRFormat ()</td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>Returns</td>
<td>the best texture format for storage of HDR data for the active device.</td>
</tr>
</tbody>
</table>
| Point3F | getDesktopResolution ()

Returns the width, height, and bitdepth of the screen/desktop. |
| string | getDisplayDeviceInformation ()

Get the string describing the active GFX device. |
| String | getDisplayDeviceList ()

Returns a tab-separated string of the detected devices across all adapters. |
| float | getPixelShaderVersion ()

Returns the pixel shader version for the active device. |
| String | getTextureProfileStats ()

Returns a list of texture profiles in the format: ProfileName TextureCount TextureMB. |
| void | listGFXResources (bool unflaggedOnly=false)

Returns a list of the unflagged GFX resources. See flagCurrentGFXResources for usage details. |
| void | reloadTextures ()

Reload all the textures from disk. |
| void | screenShot (string file, string format, int tileCount=1, float tileOverlap=0)

Takes a screenshot with optional tiling to produce huge screenshots. |
| void | setPixelShaderVersion (float version)

Sets the pixel shader version for the active device. This can be used to force a lower pixel shader version than is supported by the device for testing or performance optimization. |
| void | setReflectFormat (GFXFormat format)

Set the reflection texture format. |
<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>$gfx::disableOcclusionQuery</td>
<td>Debug helper that disables all hardware occlusion queries causing them to return only the visible state.</td>
</tr>
<tr>
<td>bool</td>
<td>$pref::Video::disableVerticalSync</td>
<td>Disables vertical sync on the active device.</td>
</tr>
<tr>
<td>bool</td>
<td>$gfx::disassembleAllShaders</td>
<td>On supported devices this will dump shader disassembly to the procedural shader folder.</td>
</tr>
<tr>
<td>float</td>
<td>$pref::Video::forcedPixVersion</td>
<td>Will force the shader model if the value is positive and less than the shader model supported by the active device. Use 0 for fixed function.</td>
</tr>
<tr>
<td>int</td>
<td>$pref::Video::textureReductionLevel</td>
<td>The number of mipmap levels to drop on loaded textures to reduce video memory usage. It will skip any textures that have been defined as not allowing down scaling.</td>
</tr>
<tr>
<td>bool</td>
<td>$gfx::wireframe</td>
<td>Used to toggle wireframe rendering at runtime.</td>
</tr>
</tbody>
</table>
Detailed Description

The low level graphics interface to the engine.

In Torque the GFX layer provides access to abstracted low level graphics concepts. From script you have limited access to graphics rendering as it is usually too slow to do individual draw calls thru the scripting interface. For drawing its usually better to use the higher level gameplay objects.

Note:

Detailed technical descriptions of when to use specific GFXStateBlockData fields, how GFXBlendOp works, or other interfaces of that nature are outside the scope of this manual. Since Torque is based on DirectX and OpenGL any reference documents for those APIs will provide the background needed to learn about rendering.
Enumeration Type Documentation

enum GFXAdapterType

Back-end graphics API used by the GFX subsystem.

Enumerator:

- **OpenGL**  OpenGL.
- **D3D8**    Direct3D 8.
- **D3D9**    Direct3D 9.
- **NullDevice**  Null device for dedicated servers.
- **Xenon**  Direct3D 9 on Xbox 360.

enum GFXBlend

The supported blend modes.

Enumerator:

- **GFXBlendZero**  (0, 0, 0, 0)
- **GFXBlendOne**   (1, 1, 1, 1)
- **GFXBlendSrcColor**  (Rs, Gs, Bs, As)
GFXBlendInvSrcColor (1 - Rs, 1 - Gs, 1 - Bs, 1 - As)

GFXBlendSrcAlpha (As, As, As, As)

GFXBlendInvSrcAlpha (1 - As, 1 - As, 1 - As, 1 - As)

GFXBlendDestAlpha (Ad Ad Ad Ad)

GFXBlendInvDestAlpha (1 - Ad 1 - Ad 1 - Ad 1 - Ad)

GFXBlendDestColor (Rd, Gd, Bd, Ad)

GFXBlendInvDestColor (1 - Rd, 1 - Gd, 1 - Bd, 1 - Ad)

GFXBlendSrcAlphaSat (f, f, f, 1) where f = \text{min}(As, 1 - Ad)

\textbf{enum GFXBlendOp}

The blend operators.

\textbf{Enumerator:}

\begin{itemize}
  \item GFXBlendOpAdd
  \item GFXBlendOpSubtract
  \item GFXBlendOpRevSubtract
  \item GFXBlendOpMin
  \item GFXBlendOpMax
\end{itemize}

\textbf{enum GFXCmpFunc}
The supported comparison functions.

**Enumerator:**

- `GFXCmpNever`
- `GFXCmpLess`
- `GFXCmpEqual`
- `GFXCmpLessEqual`
- `GFXCmpGreater`
- `GFXCmpNotEqual`
- `GFXCmpGreaterEqual`
- `GFXCmpAlways`

```c
enum GFXCullMode
```

The render cull modes.

**Enumerator:**

- `GFXCullNone`
- `GFXCullCW`
- `GFXCullCCW`

```c
enum GFXFormat
```

The texture formats.

**Note:**

Not all formats are supported on all platforms.

**Enumerator:**

- `GFXFormatR8G8B8`
- `GFXFormatR8G8B8A8`
- `GFXFormatR8G8B8X8`
- `GFXFormatR32F`
- `GFXFormatR5G6B5`
GFXFormatR5G5B5A1
GFXFormatR5G5B5X1
GFXFormatA4L4
GFXFormatA8L8
GFXFormatA8
GFXFormatL8
GFXFormatDXT1
GFXFormatDXT2
GFXFormatDXT3
GFXFormatDXT4
GFXFormatDXT5
GFXFormatD32
GFXFormatD24X8
GFXFormatD24S8
GFXFormatD24FS8
GFXFormatD16
GFXFormatR32G32B32A32F
GFXFormatR16G16B16A16F
GFXFormatL16
GFXFormatR16G16B16A16
GFXFormatR16G16
GFXFormatR16F
GFXFormatR16G16F
GFXFormatR10G10B10A2

```
enum GFXStencilOp

The stencil operators.

Enumerator:

GFXStencilOpKeep
GFXStencilOpZero
GFXStencilOpReplace
GFXStencilOpIncrSat
GFXStencilOpDecrSat
```
GFXStencilOpInvert
GFXStencilOpIncr
GFXStencilOpDecr

enum GFXTextureAddressMode

The texture address modes.

Enumerator:

GFXAddressWrap
GFXAddressMirror
GFXAddressClamp
GFXAddressBorder
GFXAddressMirrorOnce

enum GFXTextureArgument

The texture arguments.

Enumerator:

GFXTADiffuse
GFXTACurrent
GFXTATexture
GFXTATFactor
GFXTASpecular
GFXTATemp
GFXTAConstant
OneMinus
AlphaReplicate

enum GFXTextureFilterType

The texture filter types.
**Enumerator:**
- GFXTextureFilterNone
- GFXTextureFilterPoint
- GFXTextureFilterLinear
- GFXTextureFilterAnisotropic
- GFXTextureFilterPyramidalQuad
- GFXTextureFilterGaussianQuad

**Enum GFXTextureOp**

The texture operators.

**Enumerator:**
- GFXTOPDisable
- GFXTOPSelectARG1
- GFXTOPSelectARG2
- GFXTOPModulate
- GFXTOPModulate2X
- GFXTOPModulate4X
- GFXTOPAdd
- GFXTOPAddSigned
- GFXTOPAddSigned2X
- GFXTOPSubtract
- GFXTOPAddSmooth
- GFXTOPBlendDiffuseAlpha
- GFXTOPBlendTextureAlpha
- GFXTOPBlendFactorAlpha
- GFXTOPBlendTextureAlphaPM
- GFXTOPBlendCURRENTALPHA
- GFXTOPPreModulate
- GFXTOPModulateAlphaAddColor
- GFXTOPModulateColorAddAlpha
- GFXTOPModulateInvAlphaAddColor
- GFXTOPModulateInvColorAddAlpha
enum GFXTextureTransformFlags

The texture transform state flags.

**Enumerator:**

- **GFXTTFDisable**
- **GFXTTFFCoord1D**
- **GFXTTFFCoord2D**
- **GFXTTFFCoord3D**
- **GFXTTFFCoord4D**
- **GFXTTFFProjected**

enum MaterialAnimType

The type of animation effect to apply to this material.

**Enumerator:**

- **Scroll** Scroll the material along the X/Y axis.
- **Rotate** Rotate the material around a point.
- **Wave** Warps the material with an animation using Sin, Triangle or Square mathematics.
- **Scale** Scales the material larger and smaller with a pulsing effect.
**Sequence** Enables the material to have multiple frames of animation in its imagemap.

<table>
<thead>
<tr>
<th><strong>enum MaterialBlendOp</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>The type of graphical blending operation to apply to this material.</td>
</tr>
<tr>
<td><strong>Enumerator:</strong></td>
</tr>
<tr>
<td><strong>None</strong> Disable blending for this material.</td>
</tr>
<tr>
<td><strong>Mul</strong> Multiplicative blending.</td>
</tr>
<tr>
<td><strong>Add</strong> Adds the color of the material to the frame buffer with full alpha for each pixel.</td>
</tr>
<tr>
<td><strong>AddAlpha</strong> The color is modulated by the alpha channel before being added to the frame buffer.</td>
</tr>
<tr>
<td><strong>Sub</strong> Subtractive Blending. Reverses the color model, causing dark colors to have a stronger visual effect.</td>
</tr>
<tr>
<td><strong>LerpAlpha</strong> Linearly interpolates between Material color and frame buffer color based on alpha.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>enum MaterialWaveType</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>When using the Wave material animation, one of these Wave</td>
</tr>
</tbody>
</table>
Types will be used to determine the type of wave to display.

**Enumerator:**

*Sin*  Warps the material along a curved Sin Wave.

*Triangle*  Warps the material along a sharp Triangle Wave.

*Square*  Warps the material along a wave which transitions between two opposite states. As a Square Wave, the transition is quick and sudden.
### Function Documentation

**void cleanupTexturePool()**

Release the unused pooled textures in texture manager freeing up video memory.

**void clearGFXResourceFlags()**

Clears the flagged state on all allocated GFX resources. See flagCurrentGFXResources for usage details.

**See also:**
- flagCurrentGFXResources
- listGFXResources
- describeGFXResources

**void describeGFXResources(string resourceTypes, string filePath, bool unflaggedOnly = false)***

Dumps a description of GFX resources to a file or the console.

**Parameters:**

- **resourceTypes**
  A space seperated list of resource types or an empty string for all resources.

- **filePath**
  A file to dump the list to or an empty string to write to the console.

- **unflaggedOnly**
  If true only unflagged resources are dumped. See flagCurrentGFXResources.

**Note:**

The resource types can be one or more of the following:
• texture
• texture target
• window target
• vertex buffers
• primitive buffers
• fences
• cubemaps
• shaders
• stateblocks

void describeGFXStateBlocks(string filePath)

Dumps a description of all state blocks.

Parameters:

filePath  A file to dump the state blocks to or an empty string to write to the console.

void dumpRandomNormalMap()

Creates a 64x64 normal map texture filled with noise. The texture is saved to randNormTex.png in the location of the game executable.

void dumpTextureObjects()

Dumps a list of all active texture objects to the console.

Note:
This function is only available in debug builds.

void flagCurrentGFXResources()
Flags all currently allocated GFX resources. Used for resource allocation and leak tracking by flagging current resources then dumping a list of unflagged resources at some later point in execution.

**See also:**
- listGFXResources, clearGFXResourceFlags,
- describeGFXResources

```c
void flushTextureCache()
```
 Releases all textures and resurrects the texture manager.

```c
static int GFXInit::getAdapterCount()
```
 Return the number of graphics adapters available.

```c
GFXFormat getBestHDRFormat()
```
 Returns the best texture format for storage of HDR data for the active device.

```c
Point3F getDesktopResolution()
```
 Returns the width, height, and bitdepth of the screen/desktop.

```c
string getDisplayDeviceInformation()
```
 Get the string describing the active GFX device.
String getDisplayDeviceList( )

Returns a tab-separated string of the detected devices across all adapters.

float getPixelShaderVersion( )

Returns the pixel shader version for the active device.

String getTextureProfileStats( )

Returns a list of texture profiles in the format: ProfileName TextureCount TextureMB.

void listGFXResources(bool unflaggedOnly = false)

Returns a list of the unflagged GFX resources. See flagCurrentGFXResources for usage details.

See also:
flagCurrentGFXResources, clearGFXResourceFlags, describeGFXResources

void reloadTextures( )

Reload all the textures from disk.

void screenshot(string file, string format, int tileCount = 1, float tileOverlap = 0)
Takes a screenshot with optional tiling to produce huge screenshots.

**Parameters:**

- `file` The output image file path.
- `format` Either JPEG or PNG.
- `tileCount` If greater than 1 will tile the current screen size to take a large format screenshot.
- `tileOverlap` The amount of horizontal and vertical overlap between the tiles used to remove tile edge artifacts from post effects.

```c
void setPixelShaderVersion (float version )
```

Sets the pixel shader version for the active device. This can be used to force a lower pixel shader version than is supported by the device for testing or performance optimization.

**Parameters:**

- `version` The floating point shader version number.

**Note:**

This will only affect shaders/materials created after the call and should be used before the game begins.

**See also:**

- `$pref::Video::forcedPixVersion`

```c
void setReflectFormat (GFXFormat format )
```

Set the reflection texture format.
# Variable Documentation

<table>
<thead>
<tr>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool $gfx::disableOcclusionQuery</td>
</tr>
<tr>
<td>bool $pref::Video::disableVerticalSync</td>
</tr>
<tr>
<td>bool $gfx::disassembleAllShaders</td>
</tr>
<tr>
<td>float $pref::Video::forcedPixVersion</td>
</tr>
<tr>
<td>int $pref::Video::textureReductionLevel</td>
</tr>
</tbody>
</table>

**bool $gfx::disableOcclusionQuery**

Debug helper that disables all hardware occlusion queries causing them to return only the visible state.

**bool $pref::Video::disableVerticalSync**

Disables vertical sync on the active device.

**Note:**

The video mode must be reset for the change to take affect.

**bool $gfx::disassembleAllShaders**

On supported devices this will dump shader disassembly to the procedural shader folder.

**float $pref::Video::forcedPixVersion**

Will force the shader model if the value is positive and less than the shader model supported by the active device. Use 0 for fixed function.

**Note:**

The graphics device must be reset for the change to take affect.

**int $pref::Video::textureReductionLevel**
The number of mipmap levels to drop on loaded textures to reduce video memory usage. It will skip any textures that have been defined as not allowing down scaling.

```plaintext
bool $gfx::wireframe
```

Used to toggle wireframe rendering at runtime.
Materials
[GFX]

Classes, structures, functions, and variables related to Torque 3D's material system. More...
### Classes

<table>
<thead>
<tr>
<th>class</th>
<th>CustomMaterial Material object which provides more control over surface properties. More...</th>
</tr>
</thead>
</table>
### Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>void</strong> addMaterialMapping (string texName, string matName)**</td>
<td>Maps the given texture to the given material.</td>
</tr>
<tr>
<td><strong>string</strong> getMaterialMapping (string texName)**</td>
<td>Returns the name of the material mapped to this texture.</td>
</tr>
</tbody>
</table>
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>$pref::Video::defaultAnisotropy</td>
<td>Global variable defining the default anisotropy value.</td>
</tr>
<tr>
<td>void</td>
<td>dumpMaterialInstances</td>
<td>Dumps a formatted list of currently allocated material instances to the console.</td>
</tr>
<tr>
<td>void</td>
<td>reInitMaterials</td>
<td>Flushes all procedural shaders and re-initializes all active material instances.</td>
</tr>
</tbody>
</table>
Detailed Description

Classes, structures, functions, and variables related to Torque 3D's material system.
Function Documentation

```c
void addMaterialMapping(string texName, string matName)
```

Maps the given texture to the given material.

Generates a console warning before overwriting.

**Material** maps are used by terrain and interiors for triggering effects when an object moves onto a terrain block or interior surface using the associated texture.

```c
string getMaterialMapping(string texName)
```

Returns the name of the material mapped to this texture.

If no materials are found, an empty string is returned.

**Parameters:**

- `texName` Name of the texture
Variable Documentation

| int $pref::Video::defaultAnisotropy |

Global variable defining the default anisotropy value.

Controls the default anisotropic texture filtering level for all materials, including the terrain. This value can be changed at runtime to see its affect without reloading.

| void dumpMaterialInstances |

Dumps a formatted list of currently allocated material instances to the console.

| void reInitMaterials |

Flushes all procedural shaders and re-initializes all active material instances.
Shaders

[GFX]

Classes, structures, functions, and variables related to Torque 3D's shader system. More...
## Classes

<table>
<thead>
<tr>
<th>class</th>
<th>ShaderData</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Special type of data block that stores information about a handwritten shader. <a href="#">More...</a></td>
</tr>
</tbody>
</table>
Detailed Description

Classes, structures, functions, and variables related to Torque 3D's shader system.
Lighting
[Rendering]

The script functionality related to the lighting systems and lights.
More...
## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AdvancedLightBinManager</strong></td>
<td>Rendering Manager responsible for lighting, shadows, and global variables affecting both. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>LightAnimData</strong></td>
<td>A datablock which defines and performs light animation, such as rotation, brightness fade, and colorization. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>LightBase</strong></td>
<td>This is the base class for light objects. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>LightDescription</strong></td>
<td>A helper datablock used by classes (such as shapebase) that submit lights to the scene but do not use actual &quot;LightBase&quot; objects. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>LightFlareData</strong></td>
<td>Defines a light flare effect usable by scene lights. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>PointLight</strong></td>
<td>Lighting object that radiates light in all directions. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>SpotLight</strong></td>
<td>Lighting object which emits conical light in a direction. <a href="#">More...</a></td>
</tr>
</tbody>
</table>
## Modules

<table>
<thead>
<tr>
<th>Advanced Lighting</th>
</tr>
</thead>
<tbody>
<tr>
<td>The script functionality related to the Advanced Lighting system.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Basic Lighting</th>
</tr>
</thead>
<tbody>
<tr>
<td>The script functionality related to the Basic Lighting system.</td>
</tr>
</tbody>
</table>
## Functions

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>string</code></td>
<td><code>getActiveLightManager()</code></td>
<td>Returns the active light manager name.</td>
</tr>
<tr>
<td><code>String</code></td>
<td><code>getLightManagerNames()</code></td>
<td>Returns a tab seperated list of light manager names.</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><code>lightScene</code> (string completeCallbackFn=NULL, string mode=NULL)</td>
<td>Will generate static lighting for the scene if supported by the active light manager.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>resetLightManager()</code></td>
<td>Deactivates and then activates the currently active light manager. This causes most shaders to be regenerated and is often used when global rendering changes have occurred.</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><code>setLightManager</code> (string name)</td>
<td>Finds and activates the named light manager.</td>
</tr>
</tbody>
</table>
### Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>$Light::renderLightFrustums</td>
<td>Toggles rendering of light frustums when the light is selected in the editor.</td>
</tr>
<tr>
<td>bool</td>
<td>$Light::renderViz</td>
<td>Toggles visualization of light object's radius or cone.</td>
</tr>
</tbody>
</table>
## Callbacks

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void onLightManagerActivate (string name)</code></td>
<td>A callback called by the engine when a light manager is activated.</td>
</tr>
<tr>
<td><code>void onLightManagerDeactivate (string name)</code></td>
<td>A callback called by the engine when a light manager is deactivated.</td>
</tr>
</tbody>
</table>
Detailed Description

The script functionality related to the lighting systems and lights.
**Function Documentation**

```c
string getActiveLightManager();
```

Returns the active light manager name.

```c
String getLightManagerNames();
```

Returns a tab seperated list of light manager names.

```c
bool lightScene(string completeCallbackFn = NULL, string mode = NULL);
```

Will generate static lighting for the scene if supported by the active light manager.

If mode is "forceAlways", the lightmaps will be regenerated regardless of whether lighting cache files can be written to. If mode is "forceWritable", then the lightmaps will be regenerated only if the lighting cache files can be written.

**Parameters:**

- `completeCallbackFn` The name of the function to execute when the lighting is complete.
- `mode` One of "forceAlways", "forceWritable" or "loadOnly".

**Returns:**

Returns true if the scene lighting process was started.

```c
void onLightManagerActivate(string name);
```
A callback called by the engine when a light manager is activated.

**Parameters:**

*name* The name of the light manager being activated.

```c
void onLightManagerDeactivate(string name)
```

A callback called by the engine when a light manager is deactivated.

**Parameters:**

*name* The name of the light manager being deactivated.

```c
void resetLightManager()
```

Deactivates and then activates the currently active light manager. This causes most shaders to be regenerated and is often used when global rendering changes have occurred.

```c
bool setLightManager(string name)
```

Finds and activates the named light manager.

**Returns:**

Returns true if the light manager is found and activated.
Variable Documentation

bool $Light::renderLightFrustums

Toggles rendering of light frustums when the light is selected in the editor.

Note:
Only works for shadow mapped lights.

bool $Light::renderViz

Toggles visualization of light object's radius or cone.

Copyright © GarageGames, LLC. All Rights Reserved.
Advanced Lighting
[Lighting]

The script functionality related to the Advanced Lighting system.
More...
### Enumerations

```c
enum ShadowFilterMode {
    None,
    SoftShadow,
    SoftShadowHighQuality
}
```

The shadow filtering modes for Advanced Lighting shadows.

More...

```c
enum ShadowType {
    Spot,
    PSSM,
    Paraboloid,
    DualParaboloidSinglePass,
    DualParaboloid,
    CubeMap
}
```
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>$pref::PSSM::detailAdjustScale</td>
<td>Scales the model LOD when rendering into the PSSM shadow. Use this to reduce the draw calls when rendering the shadow by having meshes LOD out nearer to the camera than normal.</td>
</tr>
<tr>
<td>bool</td>
<td>$Shadows::disable</td>
<td>Used by the editor to disable all shadow rendering.</td>
</tr>
<tr>
<td>bool</td>
<td>$pref::Shadows::disable</td>
<td>Used to disable all shadow rendering.</td>
</tr>
<tr>
<td>ShadowFilterMode</td>
<td>$pref::shadows::filterMode</td>
<td>The filter mode to use for shadows.</td>
</tr>
<tr>
<td>bool</td>
<td>$AL::PSSMDebugRender</td>
<td>Enables debug rendering of the PSSM shadows.</td>
</tr>
<tr>
<td>float</td>
<td>$pref::PSSM::smallestVisiblePixelSize</td>
<td>The smallest pixel size an object can be and still be rendered into the PSSM shadow. Use this to force culling of small objects which contribute little to the final shadow.</td>
</tr>
<tr>
<td>float</td>
<td>$pref::Shadows::textureScalar</td>
<td>Used to scale the shadow texture sizes. This can reduce the shadow quality and texture memory overhead or increase them.</td>
</tr>
<tr>
<td>bool</td>
<td>$AL::UseSSAOMask</td>
<td>Used by the SSAO PostEffect to toggle the sampling of ssaomask texture by the light shaders.</td>
</tr>
</tbody>
</table>
Detailed Description

The script functionality related to the Advanced Lighting system.
### Enumeration Type Documentation

<table>
<thead>
<tr>
<th>Enum Type</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>enum ShadowFilterMode</strong></td>
<td></td>
</tr>
<tr>
<td><strong>None</strong></td>
<td>Simple point sampled filtering. This is the fastest and lowest quality mode.</td>
</tr>
<tr>
<td><strong>SoftShadow</strong></td>
<td>A variable tap rotated poisson disk soft shadow filter. It performs 4 taps to classify the point as in shadow, out of shadow, or along a shadow edge. Samples on the edge get an additional 8 taps to soften them.</td>
</tr>
<tr>
<td><strong>SoftShadowHighQuality</strong></td>
<td>A 12 tap rotated poisson disk soft shadow filter. It performs all the taps for every point without any early rejection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Enum Type</th>
<th>Enumerator</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>enum ShadowType</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Spot</strong></td>
<td></td>
</tr>
<tr>
<td><strong>PSSM</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Paraboloid</strong></td>
<td></td>
</tr>
<tr>
<td><strong>DualParaboloidSinglePass</strong></td>
<td></td>
</tr>
<tr>
<td><strong>DualParaboloid</strong></td>
<td></td>
</tr>
<tr>
<td><strong>CubeMap</strong></td>
<td></td>
</tr>
</tbody>
</table>
Variable Documentation

float $pref::PSSM::detailAdjustScale

Scales the model LOD when rendering into the PSSM shadow. Use this to reduce the draw calls when rendering the shadow by having meshes LOD out nearer to the camera than normal.

See also:

$pref::TS::detailAdjust

bool $Shadows::disable

Used by the editor to disable all shadow rendering.

bool $pref::Shadows::disable

Used to disable all shadow rendering.

ShadowFilterMode $pref::shadows::filterMode

The filter mode to use for shadows.

bool $AL::PSSMDebugRender

Enables debug rendering of the PSSM shadows.

float $pref::PSSM::smallestVisiblePixelSize

The smallest pixel size an object can be and still be rendered into
the PSSM shadow. Use this to force culling of small objects which contribute little to the final shadow.

**See also:**

$\texttt{pref::TS::smallestVisiblePixelSize}$

---

**float $\texttt{pref::Shadows::textureScalar}$**

Used to scale the shadow texture sizes. This can reduce the shadow quality and texture memory overhead or increase them.

---

**bool $\texttt{AL::UseSSAOMask}$**

Used by the SSAO **PostEffect** to toggle the sampling of ssaomask texture by the light shaders.
Basic Lighting
[Lighting]

The script functionality related to the Basic Lighting system. More...
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><code>$BasicLightManagerStats::activePlugins</code></td>
<td>The number of active Basic Lighting SceneObjectLightingPlugin objects this frame.</td>
</tr>
<tr>
<td>int</td>
<td><code>$BasicLightManagerStats::elapsedUpdateMs</code></td>
<td>The number of milliseconds spent this frame updating Basic Lighting shadows.</td>
</tr>
<tr>
<td>float</td>
<td><code>$pref::ProjectedShadow::fadeEndPixelSize</code></td>
<td>A size in pixels at which BL shadows are fully faded out. This should be a smaller value than fadeStartPixelSize.</td>
</tr>
<tr>
<td>float</td>
<td><code>$pref::ProjectedShadow::fadeStartPixelSize</code></td>
<td>A size in pixels at which BL shadows begin to fade out. This should be a larger value than fadeEndPixelSize.</td>
</tr>
<tr>
<td>float</td>
<td><code>$BasicLightManager::shadowFilterDistance</code></td>
<td>The maximum distance in meters that projected shadows will get soft filtering.</td>
</tr>
<tr>
<td>int</td>
<td><code>$BasicLightManagerStats::shadowsUpdated</code></td>
<td>The number of Basic Lighting shadows updated this frame.</td>
</tr>
</tbody>
</table>
Detailed Description

The script functionality related to the Basic Lighting system.
Variable Documentation

```plaintext
int $BasicLightManagerStats::activePlugins

The number of active Basic Lighting SceneObjectLightingPlugin objects this frame.

int $BasicLightManagerStats::elapsedUpdateMs

The number of milliseconds spent this frame updating Basic Lighting shadows.

float $pref::ProjectedShadow::fadeEndPixelSize

A size in pixels at which BL shadows are fully faded out. This should be a smaller value than fadeStartPixelSize.

See also:
DecalData

float $pref::ProjectedShadow::fadeStartPixelSize

A size in pixels at which BL shadows begin to fade out. This should be a larger value than fadeEndPixelSize.

See also:
DecalData

float $BasicLightManager::shadowFilterDistance

The maximum distance in meters that projected shadows will get...
soft filtering.

int $BasicLightManagerStats::shadowsUpdated

The number of Basic Lighting shadows updated this frame.
Render Binning
[Rendering]

The render sorting and batching system. More...
## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RenderBinManager</td>
<td>The abstract base for all render bins. <a href="#">More...</a></td>
</tr>
<tr>
<td>RenderGlowMgr</td>
<td>A render bin for the glow pass. <a href="#">More...</a></td>
</tr>
<tr>
<td>RenderImposterMgr</td>
<td>A render bin for batch rendering imposters. <a href="#">More...</a></td>
</tr>
<tr>
<td>RenderMeshMgr</td>
<td>A render bin for mesh rendering. <a href="#">More...</a></td>
</tr>
<tr>
<td>RenderObjectMgr</td>
<td>A render bin which uses object callbacks for rendering. <a href="#">More...</a></td>
</tr>
<tr>
<td>RenderOcclusionMgr</td>
<td>A render bin which renders occlusion query requests. <a href="#">More...</a></td>
</tr>
<tr>
<td>RenderParticleMgr</td>
<td>A render bin which renders particle geometry. <a href="#">More...</a></td>
</tr>
<tr>
<td>RenderPassManager</td>
<td>A grouping of render bin managers which forms a render pass. <a href="#">More...</a></td>
</tr>
<tr>
<td>RenderPassStateBin</td>
<td>A non-rendering render bin used to enable/disable a RenderPassStateToken. <a href="#">More...</a></td>
</tr>
<tr>
<td>RenderPassStateToken</td>
<td>Abstract base class for RenderFormatToken, used to manipulate what goes on in the render manager. <a href="#">More...</a></td>
</tr>
<tr>
<td>RenderPrePassMgr</td>
<td>The render bin which performs a z+normals prepass used in Advanced Lighting. <a href="#">More...</a></td>
</tr>
<tr>
<td>RenderTerrainMgr</td>
<td></td>
</tr>
</tbody>
</table>
A render bin for terrain mesh rendering.  

<table>
<thead>
<tr>
<th>class</th>
<th>RenderTexTargetBinManager</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>An abstract base class for render bin managers that render to a named texture target.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class</th>
<th>RenderTranslucentMgr</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A render bin for rendering translucent meshes.</td>
</tr>
</tbody>
</table>
Enumerations

```c
enum RenderTexTargetSize {
    windowsize,
    windowsizescaled,
    fixedsize
}
```

What size to render the target texture. Sizes are based on the Window the render is occurring in.

More...
### Variables

<table>
<thead>
<tr>
<th>static bool</th>
<th>RenderOcclusionMgr::debugRender</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A debugging feature which renders the occlusion volumes to the scene.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static bool</th>
<th>RenderTerrainMgr::renderWireframe</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Used to enable wireframe rendering on terrain for debugging.</td>
</tr>
</tbody>
</table>
Detailed Description

The render sorting and batching system.

In Torque we use a binning system to do the initial ordering and batching of rendering operations.

When rendering a pass is made thru all the game objects visible in the scene. The game objects will each submit one or more RenderInst to the RenderPassManager. The pass manager maintains an ordered list of RenderBinManagers each which get a chance to consume the RenderInst.

After all the game objects have been processed the RenderPassManager lets each bin sort then render the RenderInsts they contain.

Currently from script you can only define and change the order of the bins in the RenderPassManager. To create new types of bins or add new rendering methods you will need C++ source access.

See core\scripts\client\renderManager.cs.
### Enumeration Type Documentation

<table>
<thead>
<tr>
<th>enum</th>
<th>RenderTexTargetSize</th>
</tr>
</thead>
</table>

What size to render the target texture. Sizes are based on the Window the render is occurring in.

**Enumerator:**

- `windowsize` Render to the size of the window.
- `windowsizesscaled` Render to the size of the window, scaled to the render target's size.
- `fixedsize` Don't scale the target texture, and render to its default size.
Variable Documentation

bool RenderOcclusionMgr::debugRender [static, inherited]

A debugging feature which renders the occlusion volumes to the scene.

See also:
    RenderOcclusionMgr

bool RenderTerrainMgr::renderWireframe [static, inherited]

Used to enable wireframe rendering on terrain for debugging.

Copyright © GarageGames, LLC. All Rights Reserved.
Sound

A broad range of functionality for creating rich game audio. More...
## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SFXAmbience</strong></td>
<td>A datablock that describes an ambient sound space.</td>
</tr>
<tr>
<td><strong>SFXController</strong></td>
<td>A sound source that drives multi-source playback.</td>
</tr>
<tr>
<td><strong>SFXDescription</strong></td>
<td>A description for how a sound should be played.</td>
</tr>
<tr>
<td><strong>SFXEmitter</strong></td>
<td>An invisible 3D object that emits sound.</td>
</tr>
<tr>
<td><strong>SFXEnvironment</strong></td>
<td>Description of a reverb environment.</td>
</tr>
<tr>
<td><strong>SFXParameter</strong></td>
<td>A sound channel value that can be bound to multiple sound sources.</td>
</tr>
<tr>
<td><strong>SFXPlayList</strong></td>
<td>A datablock describing a playback pattern of sounds.</td>
</tr>
<tr>
<td><strong>SFXProfile</strong></td>
<td>Encapsulates a single sound file for playback by the sound system.</td>
</tr>
<tr>
<td><strong>SFXSound</strong></td>
<td>A sound controller that directly plays a single sound file.</td>
</tr>
<tr>
<td><strong>SFXSource</strong></td>
<td>Playback controller for a sound source.</td>
</tr>
<tr>
<td><strong>SFXSpace</strong></td>
<td>A volume in space that defines an ambient sound zone.</td>
</tr>
<tr>
<td><strong>SFXState</strong></td>
<td></td>
</tr>
</tbody>
</table>

More information available for each class.
A boolean switch used to modify playlist behavior.  

<table>
<thead>
<tr>
<th>class</th>
<th>SFXTrack</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Abstract base class for sound data that can be played back by the sound system.</td>
</tr>
</tbody>
</table>
Modules

**FMOD**

Functionality specific to the FMOD SFX implementation.
Enums

SFXChannel {
    Volume,
    Pitch,
    Priority,
    PositionX,
    PositionY,
    PositionZ,
    RotationX,
    RotationY,
    RotationZ,
    VelocityX,
    VelocityY,
    VelocityZ,
    ReferenceDistance,
    MaxDistance,
    ConeInsideAngle,
    ConeOutsideAngle,
    ConeOutsideVolume,
    Cursor,
    Status,
    User0,
    User1,
    User2,
    User3
}

Channels are individual properties of sound sources that may be animated over time.

More...

SFXDistanceModel {
    Linear,
    Logarithmic
}
<table>
<thead>
<tr>
<th>Type of volume distance attenuation curve.</th>
</tr>
</thead>
<tbody>
<tr>
<td>More...</td>
</tr>
<tr>
<td><strong>enum</strong></td>
</tr>
<tr>
<td>SFXPlayListLoopMode {</td>
</tr>
<tr>
<td>All,</td>
</tr>
<tr>
<td>Single</td>
</tr>
<tr>
<td>}</td>
</tr>
<tr>
<td>Playlist behavior when description is set to loop.</td>
</tr>
<tr>
<td>More...</td>
</tr>
<tr>
<td><strong>enum</strong></td>
</tr>
<tr>
<td>SFXPlayListRandomMode {</td>
</tr>
<tr>
<td>NotRandom,</td>
</tr>
<tr>
<td>StrictRandom,</td>
</tr>
<tr>
<td>OrderedRandom</td>
</tr>
<tr>
<td>}</td>
</tr>
<tr>
<td>Randomization pattern to apply to playlist slot playback order.</td>
</tr>
<tr>
<td>More...</td>
</tr>
<tr>
<td><strong>enum</strong></td>
</tr>
<tr>
<td>SFXPlayListReplayMode {</td>
</tr>
<tr>
<td>IgnorePlaying,</td>
</tr>
<tr>
<td>RestartPlaying,</td>
</tr>
<tr>
<td>KeepPlaying,</td>
</tr>
<tr>
<td>StartNew,</td>
</tr>
<tr>
<td>SkipIfPlaying</td>
</tr>
<tr>
<td>}</td>
</tr>
<tr>
<td>Behavior when hitting the play stage of a slot that is still playing from a previous cycle.</td>
</tr>
<tr>
<td>More...</td>
</tr>
<tr>
<td><strong>enum</strong></td>
</tr>
<tr>
<td>SFXPlayListStateMode {</td>
</tr>
<tr>
<td>StopWhenDeactivated,</td>
</tr>
<tr>
<td>PauseWhenDeactivated,</td>
</tr>
<tr>
<td>IgnoreWhenDeactivated</td>
</tr>
<tr>
<td>}</td>
</tr>
</tbody>
</table>
Reaction behavior when a state is changed incompatibly on a slot that has already started playing.

More...

```csharp
enum SFXPlayListTransitionMode {
    None,
    Wait,
    WaitAll,
    Stop,
    StopAll
}

Playlist behavior when transitioning in and out of individual slots.

More...

```csharp
enum SFXStatus {
    Playing,
    Stopped,
    Paused
}

Playback status of sound source.

More...
## Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>sfxCreateDevice</code> (string provider, string device, bool useHardware, int maxBuffers)</td>
<td>Try to create a new sound device using the given properties.</td>
</tr>
<tr>
<td>SFXSource</td>
<td><code>sfxCreateSource</code> (SFXTrack track)</td>
<td>Create a new source that plays the given track.</td>
</tr>
<tr>
<td>SFXSource</td>
<td><code>sfxCreateSource</code> (SFXTrack track, float x, float y, float z)</td>
<td>Create a new source that plays the given track and position its 3D sounds source at the given coordinates (if it is a 3D sound).</td>
</tr>
<tr>
<td>SFXSound</td>
<td><code>sfxCreateSource</code> (SFXDescription description, string filename)</td>
<td>Create a temporary SFXProfile from the given description and filename and then create and return a new source that plays the profile.</td>
</tr>
<tr>
<td>SFXSound</td>
<td><code>sfxCreateSource</code> (SFXDescription description, string filename, float x, float y, float z)</td>
<td>Create a temporary SFXProfile from the given description and filename and then create and return a new source that plays the profile. Position the sound source at the given coordinates (if it is a 3D sound).</td>
</tr>
<tr>
<td>void</td>
<td><code>sfxDeleteDevice</code> ()</td>
<td>Delete the currently active sound device and release all its resources.</td>
</tr>
<tr>
<td>void</td>
<td><code>sfxDeleteWhenStopped</code> (SFXSource source)</td>
<td>Mark the given source for deletion as soon as it moves into stopped state.</td>
</tr>
<tr>
<td>void</td>
<td><code>sfxDumpSources</code> (bool includeGroups=false)</td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>string <code>sfxDumpSourcesToString(bool includeGroups=false)</code></td>
<td>Dump information about all current <code>SFXSource</code> instances to a string.</td>
<td></td>
</tr>
<tr>
<td>string <code>sfxGetActiveStates()</code></td>
<td>Return a newline-separated list of all active states.</td>
<td></td>
</tr>
<tr>
<td>string <code>sfxGetAvailableDevices()</code></td>
<td>Get a list of all available sound devices.</td>
<td></td>
</tr>
<tr>
<td>string <code>sfxGetDeviceInfo()</code></td>
<td>Return information about the currently active sound device.</td>
<td></td>
</tr>
<tr>
<td><code>SFXDistanceModel</code> <code>sfxGetDistanceModel()</code></td>
<td>Get the falloff curve type currently being applied to 3D sounds.</td>
<td></td>
</tr>
<tr>
<td>float <code>sfxGetDopplerFactor()</code></td>
<td>Get the current global doppler effect setting.</td>
<td></td>
</tr>
<tr>
<td>float <code>sfxGetRolloffFactor()</code></td>
<td>Get the current global scale factor applied to volume attenuation of 3D sounds in the logarithmic model.</td>
<td></td>
</tr>
<tr>
<td><code>SFXSource</code> <code>sfxPlay(SFXSource source)</code></td>
<td>Start playback of the given source.</td>
<td></td>
</tr>
<tr>
<td>void <code>sfxPlay(SFXTrack track)</code></td>
<td>Create a new play-once source for the given track and start playback of the source.</td>
<td></td>
</tr>
<tr>
<td>void <code>sfxPlay(SFXTrack track, float x, float y, float z)</code></td>
<td>Create a new play-once source for the given track, position its 3D sound at the given coordinates (if the track’s description is set up for 3D sound) and start playback of the</td>
<td></td>
</tr>
</tbody>
</table>
source.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SFXSource</strong> sfxPlayOnce (SFXTrack track)</td>
<td>Create a play-once source for the given track.</td>
</tr>
<tr>
<td><strong>SFXSource</strong> sfxPlayOnce (SFXTrack track, float x, float y, float z, float fadeInTime=-1)</td>
<td>Create a play-once source for the given given track and position the source's 3D sound at the given coordinates only if the track's description is set up for 3D sound.</td>
</tr>
<tr>
<td><strong>SFXSource</strong> sfxPlayOnce (SFXDescription description, string filename)</td>
<td>Create a new temporary SFXProfile from the given description and filename, then create a play-once source for it and start playback.</td>
</tr>
<tr>
<td><strong>SFXSource</strong> sfxPlayOnce (SFXDescription description, string filename, float x, float y, float z, float fadeInTime=-1)</td>
<td>Create a new temporary SFXProfile from the given description and filename, then create a play-once source for it and start playback. Position the source's 3D sound at the given coordinates (only if the description is set up for 3D sound).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void sfxSetDistanceModel (SFXDistanceModel model)</td>
<td>Set the falloff curve type to use for distance-based volume attenuation of 3D sounds.</td>
</tr>
<tr>
<td>void sfxSetDopplerFactor (float value)</td>
<td>Set the global doppler effect scale factor.</td>
</tr>
<tr>
<td>void sfxSetRolloffFactor (float value)</td>
<td>Set the global scale factor to apply to volume attenuation of 3D sounds in the logarithmic model.</td>
</tr>
<tr>
<td>void</td>
<td><code>sfxStop</code> <em>(SFXSource source)</em></td>
</tr>
<tr>
<td>--------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>void</td>
<td><code>sfxStopAndDelete</code> <em>(SFXSource source)</em></td>
</tr>
</tbody>
</table>
### Variables

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><code>$SFX::ambientUpdateTime</code>&lt;br&gt;Milliseconds spent on the last ambient audio update.</td>
</tr>
<tr>
<td>const int</td>
<td><code>$SFX::DEVICE_CAPS_DSPEFFECTS</code>&lt;br&gt;Sound device capability flag indicating that the sound device supports adding DSP effect chains to sounds.</td>
</tr>
<tr>
<td>const int</td>
<td><code>$SFX::DEVICE_CAPS_FMODDESIGNER</code>&lt;br&gt;Sound device capability flag indicating that the sound device supports FMOD Designer audio projects.</td>
</tr>
<tr>
<td>const int</td>
<td><code>$SFX::DEVICE_CAPS_MULTILISTENER</code>&lt;br&gt;Sound device capability flag indicating that the sound device supports multiple concurrent listeners.</td>
</tr>
<tr>
<td>const int</td>
<td><code>$SFX::DEVICE_CAPS_OCCLUSION</code>&lt;br&gt;Sound device capability flag indicating that the sound device implements sound occlusion.</td>
</tr>
<tr>
<td>const int</td>
<td><code>$SFX::DEVICE_CAPS_REVERB</code>&lt;br&gt;Sound device capability flag indicating that the sound device supports reverb.</td>
</tr>
<tr>
<td>const int</td>
<td><code>$SFX::DEVICE_CAPS_VOCEMANAGEMENT</code>&lt;br&gt;Sound device capability flag indicating that the sound device implements its own voice virtualization.</td>
</tr>
<tr>
<td>const int</td>
<td><code>$SFX::DEVICE_INFO_CAPS</code>&lt;br&gt;Index of device capability flags in device info string.</td>
</tr>
<tr>
<td>const int</td>
<td><code>$SFX::DEVICE_INFO_MAXBUFFERS</code>&lt;br&gt;Index of buffer limit number in device info string.</td>
</tr>
<tr>
<td>const int</td>
<td><code>$SFX::DEVICE_INFO_NAME</code>&lt;br&gt;Index of device name field in device info string.</td>
</tr>
<tr>
<td>const int</td>
<td><code>$SFX::DEVICE_INFO_PROVIDER</code>&lt;br&gt;Index of sound provider field in device info string.</td>
</tr>
</tbody>
</table>
| const int | `$SFX::DEVICE_INFO_USEHARDWARE`<br>
<table>
<thead>
<tr>
<th>Variable Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| int          | **$SFX::numCulled**  
Number of SFXSounds that are currently in virtualized playback mode. |
| int          | **$SFX::numPlaying**  
Number of SFXSources that are currently in playing state. |
| int          | **$SFX::numSounds**  
Number of SFXSound type objects (i.e. actual single-file sounds) that are currently instantiated. |
| int          | **$SFX::numSources**  
Number of SFXSource type objects that are currently instantiated. |
| int          | **$SFX::numVoices**  
Number of voices that are currently allocated on the sound device. |
| int          | **$SFX::parameterUpdateTime**  
Milliseconds spent on the last SFXParameter update loop. |
| static Color | **SFXEmitter::renderColorInnerCone**  
The color with which to render dots in the inner sound cone (Editor only). |
| static Color | **SFXEmitter::renderColorOuterCone**  
The color with which to render dots in the outer sound cone (Editor only). |
| static Color | **SFXEmitter::renderColorOutsideVolume**  
The color with which to render dots outside of the outer sound cone (Editor only). |
| static Color | **SFXEmitter::renderColorPlayingInRange**  
The color with which to render a sound emitter's marker cube in the editor when the emitter's sound is playing and in range of the listener. |
<table>
<thead>
<tr>
<th>static Colorl</th>
<th>SFXEmitter::renderColorPlayingOutOfRange</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The color with which to render a sound emitter's marker cube in the editor when the emitter's sound is playing but out of the range of the listener.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static Colorl</th>
<th>SFXEmitter::renderColorRangeSphere</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The color of the range sphere with which to render sound emitters in the editor.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static Colorl</th>
<th>SFXEmitter::renderColorStoppedInRange</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The color with which to render a sound emitter's marker cube in the editor when the emitter's sound is not playing but the emitter is in range of the listener.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static Colorl</th>
<th>SFXEmitter::renderColorStoppedOutOfRange</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The color with which to render a sound emitter's marker cube in the editor when the emitter's sound is not playing and the emitter is out of range of the listener.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static bool</th>
<th>SFXEmitter::renderEmitters</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Whether to render enhanced range feedback in the editor on all emitters regardless of selection state.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static float</th>
<th>SFXEmitter::renderPointDistance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The distance between individual points in the sound emitter rendering in the editor as the points move from the emitter's center away to maxDistance.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static float</th>
<th>SFXEmitter::renderRadialIncrements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The stepping (in degrees) for the radial sweep along the axis of the XY plane sweep for sound emitter rendering in the editor.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static float</th>
<th>SFXEmitter::renderSweepIncrements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The stepping (in degrees) for the radial sweep on the XY plane for sound emitter rendering in the editor.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int</th>
<th>$SFX::sourceUpdateTime</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Milliseconds spent on the last SFXSource update loop.</td>
</tr>
</tbody>
</table>
Detailed Description

A broad range of functionality for creating rich game audio.
3D Audio

Streaming vs. Buffered Audio

See also:

SFXDescription::isStreaming
Interactive Audio

See also:

SFXPlayList
SFXState
Ambient Audio

See also:

SFXAmbience
Audio Reverb

Please consult the EAX documentation for more information

Note:
Reverb currently only works with FMOD.

See also:
http://connect.creativelabs.com/developer/Gaming/Forms/AllItems.
SFXEnvironment
SFXDescription::useCustomReverb
Supported Sound File Formats

Note:

When using FMOD, the sound system supports all formats implemented by FMOD. Before applying file lookup patterns to a sound file path, the file name is first passed unaltered to FMOD. This allows to, for example, use URLs for internet radio playback.
Providers and Devices

- FMOD
- OpenAL
- XAudio2
- DirectSound
- Null

See also:

- sfxGetAvailableDevices
- sfxGetDeviceInfo
- sfxCreateDevice
- sfxDeleteDevice
Sound System Updates

Live Asset Updates of Sound Files
Enumeration Type Documentation

enum SFXChannel

Channels are individual properties of sound sources that may be animated over time.

See also:
- SFXParameter

Interactive Audio

Enumerator:

- **Volume**: Channel controls volume level of attached sound sources.
  
  See also:
  - SFXDescription::volume

- **Pitch**: Channel controls pitch of attached sound sources.
  
  See also:
  - SFXDescription::pitch

- **Priority**: Channel controls virtualization priority level of attached sound sources.
  
  See also:
  - SFXDescription::priority

- **PositionX**: Channel controls X coordinate of 3D sound position of attached sources.

- **PositionY**: Channel controls Y coordinate of 3D sound
position of attached sources.

**PositionZ**
Channel controls Z coordinate of 3D sound position of attached sources.

**RotationX**
Channel controls X rotation (in degrees) of 3D sound orientation of attached sources.

**RotationY**
Channel controls Y rotation (in degrees) of 3D sound orientation of attached sources.

**RotationZ**
Channel controls Z rotation (in degrees) of 3D sound orientation of attached sources.

**VelocityX**
Channel controls X coordinate of 3D sound velocity vector of attached sources.

**VelocityY**
Channel controls Y coordinate of 3D sound velocity vector of attached sources.

**VelocityZ**
Channel controls Z coordinate of 3D sound velocity vector of attached sources.

**ReferenceDistance**
Channel controls reference distance of 3D sound of attached sources.

**See also:**

*SFXDescription::referenceDistance*

**MaxDistance**
Channel controls max volume attenuation distance of 3D sound of attached sources.
See also:
  SFXDescription::maxDistance

**ConeInsideAngle**

Channel controls angle (in degrees) of 3D sound inner volume cone of attached sources.

See also:
  SFXDescription::coneInsideAngle

**ConeOutsideAngle**

Channel controls angle (in degrees) of 3D sound outer volume cone of attached sources.

See also:
  SFXDescription::coneOutsideAngle

**ConeOutsideVolume**

Channel controls volume outside of 3D sound outer cone of attached sources.

See also:
  SFXDescription::coneOutsideVolume

**Cursor**

Channel controls playback cursor of attached sound sources.

**Note:**

Be aware that different types of sound sources interpret play cursor positions differently or do not actually have play cursors (these sources will ignore the channel).

**Status**

The channel's value is rounded down to the nearest integer and interpreted in the
following way:

- 1: Play
- 2: Stop
- 3: Pause

Channel available for custom use. By default ignored by sources.

**Note:**

For FMOD Designer event sources (SFXFMODEventSource), this channel is used for event parameters defined in FMOD Designer and should not be used otherwise.

**See also:**

SFXSource::onParameterValueChange

Channel available for custom use. By default ignored by sources.

**See also:**

SFXSource::onParameterValueChange

Channel available for custom use. By default ignored by sources.

**See also:**

SFXSource::onParameterValueChange

Channel available for custom use. By default ignored by sources.

**See also:**

SFXSource::onParameterValueChange

```c
enum SFXDistanceModel
```
Type of volume distance attenuation curve.

The distance model determines the falloff curve applied to the volume of 3D sounds over distance.

**Volume Attenuation**

3D Audio

**Enumerator:**

- **Linear**
  Volume attenuates linearly from the references distance onwards to max distance where it reaches zero.

- **Logarithmic**
  Volume attenuates logarithmically starting from the reference distance and halving every reference distance step from there on. Attenuation stops at max distance but volume won't reach zero.

```
enum SFXPlayListLoopMode
```

Playlist behavior when description is set to loop.

**See also:**

- `SFXDescription::isLooping`
- `SFXPlayList::loopMode`

**Enumerator:**

- **All**
  Loop over all slots, i.e. jump from last to first slot after all slots have played.
Loop infinitely over the current slot. Only useful in combination with either states or manual playlist control.

**Single**

**Enum SFXPlayListRandomMode**

Randomization pattern to apply to playlist slot playback order.

**See also:**

*SFXPlayList::random*

**Enumerator:**

- **NotRandom**
  
  Play slots in sequential order. No randomization.

- **StrictRandom**
  
  Play a strictly random selection of slots.

  In this mode, a set of `numSlotsToPlay` random numbers between 0 and `numSlotsToPlay-1` (inclusive), i.e. in the range of valid slot indices, is generated and playlist slots are played back in the order of this list. This allows the same slot to occur multiple times in a list and, consequentially, allows for other slots to not appear at all in a given slot ordering.

- **OrderedRandom**
  
  Play all slots in the list in a random order.

  In this mode, the `numSlotsToPlay` slots from the list with valid tracks assigned are put into a random order and played. This guarantees that each slots is played exactly once albeit at a random position in the total
ordering.

```plaintext
enum SFXPlayListReplayMode

Behavior when hitting the play stage of a slot that is still playing from a previous cycle.

See also:
SFXPlayList::replay

Enumerator:

IgnorePlaying Ignore any sources that may already be playing on the slot and just create a new source.

RestartPlaying Restart all sources that was last created for the slot.

KeepPlaying Keep playing the current source(s) as if the source started last on the slot was created in this cycle. For this, the sources associated with the slot are brought to the top of the play stack.

StartNew Stop all sources currently playing on the slot and then create a new source.

SkipIfPlaying If there are sources already playing on the slot, skip the play stage.
```
enum SFXPlayListStateMode

Reaction behavior when a state is changed incompatibly on a slot that has already started playing.

See also:

SFXPlayList::stateMode

Enumerator:

StopWhenDeactivated

Stop the sources playing on the slot when a state changes to a setting that is incompatible with the slot's state setting.

PauseWhenDeactivated

Pause all sources playing on the slot when a state changes to a setting that is incompatible with the slot's state setting.

When the slot's state is reactivated again, the sources will resume playback.

IgnoreWhenDeactivated

Ignore when a state changes to a setting incompatible with the slot's state setting and just keep playing sources attached to the slot.

enum SFXPlayListTransitionMode

Playlist behavior when transitioning in and out of individual slots.

Transition behaviors apply when the playback controller starts
processing a playlist slot and when it ends processing a slot. Using transition behaviors, playback can be synchronized.

See also:
- SFXPlayList::transitionIn
- SFXPlayList::transitionOut

**Enumerator:**

- **None**
  No transition. Immediately move on to processing the slot or immediately move on to the next slot.

- **Wait**
  Wait for the sound source spawned last by this playlist to finish playing. Then proceed.

- **WaitAll**
  Wait for all sound sources currently spawned by the playlist to finish playing. Then proceed.

- **Stop**
  Stop the sound source spawned last by this playlist. Then proceed.

- **StopAll**
  Stop all sound sources spawned by the playlist. Then proceed.

```csharp
enum SFXStatus
```

Playback status of sound source.

**Enumerator:**

- **Playing**
  The source is currently playing.
**Stopped**  Playback of the source is stopped. When transitioning to Playing state, playback will start at the beginning of the source.

**Paused**  Playback of the source is paused. Resuming playback will play from the current playback position.
Function Documentation

```c
bool sfxCreateDevice(string provider,
                     string device,
                     bool  useHardware,
                     int   maxBuffers
)
```

Try to create a new sound device using the given properties.

If a sound device is currently initialized, it will be uninitialized first. However, be aware that in this case, if this function fails, it will not restore the previously active device but rather leave the sound system in an uninitialized state.

Sounds that are already playing while the new device is created will be temporarily transitioned to virtualized playback and then resume normal playback once the device has been created.

In the core scripts, sound is automatically set up during startup in the sfxStartup() function.

**Parameters:**

- `provider` The name of the device provider as returned by `sfxGetAvailableDevices()`.
- `device` The name of the device as returned by `sfxGetAvailableDevices()`.
- `useHardware` Whether to enabled hardware mixing on the device or not. Only relevant if supported by the given device.
- `maxBuffers` The maximum number of concurrent voices for this device to use or -1 for the device to pick its own reasonable default.

**Returns:**

True if the initialization was successful, false if not.
Note:
This function must be called before any of the sound playback functions can be used.

See also:
sfxGetAvailableDevices
sfxGetDeviceInfo
sfxDeleteDevice

Providers and Devices

SFXSource sfxCreateSource(SFXTrack track)

Create a new source that plays the given track.

The source will be returned in stopped state. Call SFXSource::play() to start playback.

In contrast to play-once sources, the source object will not be automatically deleted once playback stops. Call delete() to release the source object.

This function will automatically create the right SFXSource type for the given SFXTrack.

Parameters:

track The track the source should play.

Returns:
A new SFXSource for playback of the given track or 0 if no source could be created from the given track.

Note:
Trying to create a source for a device-specific track type will fail if the currently selected device does not support the type. Example: trying to create a source for an FMOD Designer event when not running FMOD.
Example:

```c
// Create and play a source from a pre-existing profile:
%source = sfxCreateSource( SoundFileProfile );
%source.play();
```

```c
SFXSource sfxCreateSource(SFXTrack track,
 float   x,
 float   y,
 float   z
)
```

Create a new source that plays the given track and position its 3D sounds source at the given coordinates (if it is a 3D sound).

The source will be returned in stopped state. Call SFXSource::play() to start playback.

In contrast to play-once sources, the source object will not be automatically deleted once playback stops. Call delete() to release the source object.

This function will automatically create the right SFXSource type for the given SFXTrack.

**Parameters:**

- `track` The track the source should play.
- `x` The X coordinate of the 3D sound position.
- `y` The Y coordinate of the 3D sound position.
- `z` The Z coordinate of the 3D sound position.

**Returns:**

A new SFXSource for playback of the given track or 0 if no source could be created from the given track.

**Note:**
Trying to create a source for a device-specific track type will fail if the currently selected device does not support the type. Example: trying to create a source for an FMOD Designer event when not running FMOD.

Example:

```
// Create and play a source from a pre-existing profile and position it at (100, 200, 300):
%source = sfxCreateSource(SoundFileProfile, 100, 200, 300);
%source.play();
```

```cpp
SFXSound sfxCreateSource(SFXDescription description, string filename)
```

Create a temporary `SFXProfile` from the given `description` and `filename` and then create and return a new source that plays the profile.

The source will be returned in stopped state. Call `SFXSource::play()` to start playback.

In contrast to play-once sources, the source object will not be automatically deleted once playback stops. Call `delete()` to release the source object.

Parameters:

- `description` The description to use for setting up the temporary `SFXProfile`.
- `filename` Path to the sound file to play.

Returns:

A new `SFXSource` for playback of the given track or 0 if no source or no temporary profile could be created.

Example:
Create a source for a music track:
%source = sfxCreateSource( AudioMusicLoop2D, 
%source.play();

See also:
SFXProfile

SFXSound sfxCreateSource(SFXDescription description,
     string filename,
     float x,
     float y,
     float z)

Create a temporary SFXProfile from the given description and filename and then create and return a new source that plays the profile. Position the sound source at the given coordinates (if it is a 3D sound).

The source will be returned in stopped state. Call SFXSource::play() to start playback.

In contrast to play-once sources, the source object will not be automatically deleted once playback stops. Call delete() to release the source object.

Parameters:

- **description**: The description to use for setting up the temporary SFXProfile.
- **filename**: Path to the sound file to play.
- **x**: The X coordinate of the 3D sound position.
- **y**: The Y coordinate of the 3D sound position.
- **z**: The Z coordinate of the 3D sound position.

Returns:
A new `SFXSource` for playback of the given track or 0 if no source or no temporary profile could be created.

**Example:**

```c
// Create a source for a music track and play it:
%source = sfxCreateSource( AudioMusicLoop3D, 0);
%source.play();
```

**See also:**
- `SFXProfile`

---

**`void sfxDeleteDevice()`**

Delete the currently active sound device and release all its resources.

SFXSources that are still playing will be transitioned to virtualized playback mode. When creating a new device, they will automatically transition back to normal playback.

In the core scripts, this is done automatically for you during shutdown in the `sfxShutdown()` function.

**See also:**
- `sfxCreateDevice`
- `Providers and Devices`

---

**`void sfxDeleteWhenStopped(SFXSource source)`**

Mark the given `source` for deletion as soon as it moves into stopped state.

This function will retroactively turn the given `source` into a play-once source (see `Play-Once Sources`).
Parameters:

source A sound source.

void sfxDumpSources(bool includeGroups = false)

Dump information about all current SFXSource instances to the console.

The dump includes information about the playback status for each source, volume levels, virtualization, etc.

Parameters:

includeGroups If true, direct instances of SFXSources (which represent logical sound groups) will be included. Otherwise only instances of subclasses of SFXSources are included in the dump.

See also:

SFXSource
sfxDumpSourcesToString

string sfxDumpSourcesToString(bool includeGroups = false)

Dump information about all current SFXSource instances to a string.

The dump includes information about the playback status for each source, volume levels, virtualization, etc.

Parameters:

includeGroups If true, direct instances of SFXSources (which represent logical sound groups) will be included. Otherwise only instances of subclasses of SFXSources are included in the
dump.

**Returns:**
A string containing a dump of information about all currently instantiated SFXSources.

**See also:**
- SFXSource
- sfxDumpSources

```c
string sfxGetActiveStates( )
```

Return a newline-separated list of all active states.

**Returns:**
A list of the form

```
stateName1 NL stateName2 NL stateName3 ...
```

where each element is the name of an active state object.

**Example:**
```
// Disable all active states.
foreach$( %state in sfxGetActiveStates() ) : %state.disable();
```

```c
string sfxGetAvailableDevices( )
```

Get a list of all available sound devices.

The return value will be a newline-separated list of entries where each line describes one available sound device. Each such line will have the following format:

```
provider TAB device TAB hasHardware TAB numMa
```
- provider: The name of the device provider (e.g. "FMOD").
- device: The name of the device as returned by the device layer.
- hasHardware: Whether the device supports hardware mixing or not.
- numMaxBuffers: The maximum number of concurrent voices supported by the device's mixer. If this limit is exceeded, i.e. if there are more active sounds playing at any one time, then voice virtualization will start culling voices and put them into virtualized playback mode. Voice virtualization may or may not be provided by the device itself; if not provided by the device, it will be provided by Torque's sound system.

Returns:
A newline-separated list of information about all available sound devices.

See also:
sfxCreateDevice
sfxGetDeviceInfo
$SFX::DEVICE_INFO_PROVIDER
$SFX::DEVICE_INFO_NAME
$SFX::DEVICE_INFO_USEHARDWARE
$SFX::DEVICE_INFO_MAXBUFFERS

Providers and Devices

string sfxGetDeviceInfo( )

Return information about the currently active sound device.

The return value is a tab-delimited string of the following format:

```
provider TAB device TAB hasHardware TAB numMaxBuffers
```

- provider: The name of the device provider (e.g. "FMOD").
- device: The name of the device as returned by the device
layer.
- hasHardware: Whether the device supports hardware mixing or not.
- numMaxBuffers: The maximum number of concurrent voices supported by the device's mixer. If this limit is exceeded, i.e. if there are more active sounds playing at any one time, then voice virtualization will start culling voices and put them into virtualized playback mode. Voice virtualization may or may not be provided by the device itself; if not provided by the device, it will be provided by Torque's sound system.
- caps: A bitfield of capability flags.

Returns:
A tab-separated list of properties of the currently active sound device or the empty string if no sound device has been initialized.

See also:
sfxCreateDevice
sfxGetAvailableDevices
$SFX::DEVICE_INFO_PROVIDER
$SFX::DEVICE_INFO_NAME
$SFX::DEVICE_INFO_USEHARDWARE
$SFX::DEVICE_INFO_MAXBUFFERS
$SFX::DEVICE_INFO_CAPS
$SFX::DEVICE_CAPS_REVERB
$SFX::DEVICE_CAPS_VOICEMANAGEMENT
$SFX::DEVICE_CAPS_OCCLUSION
$SFX::DEVICE_CAPS_DSPEFFECTS
$SFX::DEVICE_CAPS_MULTILISTENER
$SFX::DEVICE_CAPS_FMODDESIGNER

Providers and Devices

SFXDistanceModel sfxGetDistanceModel( )

Get the falloff curve type currently being applied to 3D sounds.
**Returns:**

The current distance model type.

**Volume Attenuation**

3D Audio

```c
float sfxGetDopplerFactor()
```

Get the current global doppler effect setting.

**Returns:**

The current global doppler effect scale factor (>=0).

**See also:**

`sfxSetDopplerFactor`

Doppler Effect

```c
float sfxGetRolloffFactor()
```

Get the current global scale factor applied to volume attenuation of 3D sounds in the logarithmic model.

**Returns:**

The current scale factor for logarithmic 3D sound falloff curves.

**See also:**

`sfxGetDistanceModel`

SFXDistanceModel

**Volume Attenuation 3D Audio**
**SFXSource sfxPlay(SFXSource source)**

Start playback of the given source.

This is the same as calling SFXSource::play() directly.

**Parameters:**

- `source` The source to start playing.

**Returns:**

- `source`.

**Example:**

```cpp
// Create and play a source from a pre-existing profile:
%source = sfxCreateSource( SoundFileProfile);
%source.play();
```

**void sfxPlay(SFXTrack track)**

Create a new play-once source for the given `track` and start playback of the source.

This is equivalent to calling `sfxCreateSource()` on and SFXSource::play() on the resulting source.

**Parameters:**

- `track` The sound datablock to play.

**Returns:**

- The newly created play-once source or 0 if the creation failed.

**Play-Once Sources**

**void sfxPlay(SFXTrack track,**
Create a new play-once source for the given *track*, position its 3D sound at the given coordinates (if the track's description is set up for 3D sound) and start playback of the source.

This is equivalent to calling `sfxCreateSource()` on and `SFXSource::play()` on the resulting source.

**Parameters:**

- *track* The sound datablock to play.
- *x* The X coordinate of the 3D sound position.
- *y* The Y coordinate of the 3D sound position.
- *z* The Z coordinate of the 3D sound position.

**Returns:**

The newly created play-once source or 0 if the creation failed.

**Play-Once Sources**

```cpp
SFXSource sfxPlayOnce(SFXTrack track)
```

Create a play-once source for the given *track*.

Once playback has finished, the source will be automatically deleted in the next sound system update.

**Parameters:**

- *track* The sound datablock.

**Returns:**

A newly created temporary source in "Playing" state or 0 if the operation failed.
Play-Once Sources

SFXSource sfxPlayOnce (SFXTrack track, 
    float x, 
    float y, 
    float z, 
    float fadeInTime = -1 
) 

Create a play-once source for the given track and position the source's 3D sound at the given coordinates only if the track's description is set up for 3D sound).

Once playback has finished, the source will be automatically deleted in the next sound system update.

Parameters:

- **track** The sound datablock.
- **x** The X coordinate of the 3D sound position.
- **y** The Y coordinate of the 3D sound position.
- **z** The Z coordinate of the 3D sound position.
  - If >=0, this overrides the
    - **fadeInTime** SFXDescription::fadeInTime value on the track's description.

Returns:

A newly created temporary source in "Playing" state or 0 if the operation failed.

Example:

```c
// Immediately start playing the given track
sfxPlayOnce( MusicTrack, 0, 0, 0, 5.f );
```
SFXSource sfxPlayOnce(SFXDescription description, string filename)

Create a new temporary SFXProfile from the given description and filename, then create a play-once source for it and start playback.

Once playback has finished, the source will be automatically deleted in the next sound system update. If not referenced otherwise by then, the temporary SFXProfile will also be deleted.

**Parameters:**

- **description** The description to use for playback.
- **filename** Path to the sound file to play.

**Returns:**

A newly created temporary source in "Playing" state or 0 if the operation failed.

**Example:**

```
// Play a sound effect file once.
sfxPlayOnce( AudioEffects, "art/sound/weapons/Shoot\n"
```

**Play-Once Sources**

SFXSource sfxPlayOnce(SFXDescription description, string filename,
float x,
float y,
float z,
float fadeInTime = -1)

Create a new temporary SFXProfile from the given description and filename, then create a play-once source for it and start playback.
Position the source's 3D sound at the given coordinates (only if the description is set up for 3D sound).

Once playback has finished, the source will be automatically deleted in the next sound system update. If not referenced otherwise by then, the temporary SFXProfile will also be deleted.

Parameters:

- **description** The description to use for playback.
- **filename** Path to the sound file to play.
- **x** The X coordinate of the 3D sound position.
- **y** The Y coordinate of the 3D sound position.
- **z** The Z coordinate of the 3D sound position.
- **fadeInTime** If >=0, this overrides the SFXDescription::fadeInTime value on the track's description.

Returns:

A newly created temporary source in "Playing" state or 0 if the operation failed.

Example:

```
// Play a sound effect file once using a 3D source
sfxPlayOnce( AudioDefault3D, "art/sound/weapons/Weapon_pickup" );
```

Play-Once Sources

```cpp
void sfxSetDistanceModel(SFXDistanceModel model )
```

Set the falloff curve type to use for distance-based volume attenuation of 3D sounds.

Parameters:

- **model** The distance model to use for 3D sound.

Note:
This setting takes effect globally and is applied to all 3D sounds.

```c
void sfxSetDopplerFactor(float value)
```

Set the global doppler effect scale factor.

**Parameters:**

- `value` The new doppler shift scale factor.

**Precondition:**

- `value` must be >= 0.

**See also:**

- `sfxGetDopplerFactor`

Doppler Effect

```c
void sfxSetRolloffFactor(float value)
```

Set the global scale factor to apply to volume attenuation of 3D sounds in the logarithmic model.

**Parameters:**

- `value` The new scale factor for logarithmic 3D sound falloff curves.

**Precondition:**

- `value` must be > 0.

**Note:**

- This function has no effect if the currently distance model is set to SFXDistanceModel::Linear.

**See also:**
### Volume Attenuation 3D Audio

**void sfxStop(SFXSource source )**

Stop playback of the given `source`.

This is equivalent to calling `SFXSource::stop()`.

**Parameters:**

- `source` The source to put into stopped state.

**void sfxStopAndDelete(SFXSource source )**

Stop playback of the given `source` (if it is not already stopped) and delete the `source`.

The advantage of this function over directly calling `delete()` is that it will correctly handle volume fades that may be configured on the source. Whereas calling `delete()` would immediately stop playback and delete the source, this functionality will wait for the fade-out to play and only then stop the source and delete it.

**Parameters:**

- `source` A sound source.

**Volume Fades**
Variable Documentation

**int $SFX::ambientUpdateTime**

Milliseconds spent on the last ambient audio update.

**Sound System Updates**

**Ambient Audio**

**int $SFX::DEVICE_CAPS_DSPEFFECTS**

Sound device capability flag indicating that the sound device supports adding DSP effect chains to sounds.

**See also:**

sfxGetDeviceInfo

**Note:**

This is not yet used by the sound system.

**See also:**

sfxGetDeviceInfo

**int $SFX::DEVICE_CAPS_FMODDESIGNER**

Sound device capability flag indicating that the sound device supports FMOD Designer audio projects.

**Note:**

This is exclusive to FMOD. If the FMOD Event DLLs are in place and could be successfully loaded, this flag will be set after initializating an FMOD audio device.
See also:

sfxGetDeviceInfo

**FMOD Designer Audio**

<table>
<thead>
<tr>
<th>int $SFX::DEVICE_CAPS_MULTILISTENER</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>
| Sound device capability flag indicating that the sound device supports multiple concurrent listeners.  

**Note:**

Currently only FMOD implements this.

See also:

sfxGetDeviceInfo

<table>
<thead>
<tr>
<th>int $SFX::DEVICE_CAPS_OCCLUSION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>
| Sound device capability flag indicating that the sound device implements sound occlusion.

**Note:**

This is not yet used by the sound system.

See also:

sfxGetDeviceInfo

<table>
<thead>
<tr>
<th>int $SFX::DEVICE_CAPS_REVERB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>
| Sound device capability flag indicating that the sound device supports reverb.

**Note:**
Currently only FMOD implements this.

**See also:**
- `sfxGetDeviceInfo`

Audio Reverb

```cpp
int $SFX::DEVICE_CAPS_VOICEMANAGEMENT
```

Sound device capability flag indicating that the sound device implements its own voice virtualization.

For these devices, the sound system will deactivate its own voice management and leave voice virtualization entirely to the device.

**Note:**
- Currently only FMOD implements this.

**See also:**
- `sfxGetDeviceInfo`

Sounds and Voices

```cpp
int $SFX::DEVICE_INFO_CAPS
```

Index of device capability flags in device info string.

**See also:**
- `sfxGetDeviceInfo`
- `sfxGetAvailableDevices`

```cpp
int $SFX::DEVICE_INFO_MAXBUFFERS
```

Index of buffer limit number in device info string.
### int $SFX::DEVICE_INFO_NAME

Index of device name field in device info string.

**See also:**

- sfxGetDeviceInfo
- sfxGetAvailableDevices

### int $SFX::DEVICE_INFO_PROVIDER

Index of sound provider field in device info string.

**See also:**

- sfxGetDeviceInfo
- sfxGetAvailableDevices

### int $SFX::DEVICE_INFO_USEHARDWARE

Index of use hardware flag in device info string.

**See also:**

- sfxGetDeviceInfo
- sfxGetAvailableDevices

### int $SFX::numCulled

Number of SFXSounds that are currently in virtualized playback mode.
Sounds and Voices

int $SFX::numPlaying

Number of SFXSources that are currently in playing state.

int $SFX::numSounds

Number of SFXSound type objects (i.e. actual single-file sounds) that are currently instantiated.

int $SFX::numSources

Number of SFXSource type objects that are currently instantiated.

int $SFX::numVoices

Number of voices that are currently allocated on the sound device.

int $SFX::parameterUpdateTime

Milliseconds spent on the last SFXParameter update loop.

Sound System Updates

Interactive Audio

ColorI SFXEmitter::renderColorInnerCone [static, inherited]

The color with which to render dots in the inner sound cone (Editor
<table>
<thead>
<tr>
<th>Function Name</th>
<th>Details</th>
</tr>
</thead>
</table>
| ColorI SFXEmitter::renderColorOuterCone | [static, inherited]  
The color with which to render dots in the outer sound cone (Editor only). |
| ColorI SFXEmitter::renderColorOutsideVolume | [static, inherited]  
The color with which to render dots outside of the outer sound cone (Editor only). |
| ColorI SFXEmitter::renderColorPlayingInRange | [static, inherited]  
The color with which to render a sound emitter's marker cube in the editor when the emitter's sound is playing and in range of the listener. |
| ColorI SFXEmitter::renderColorPlayingOutOfRange | [static, inherited]  
The color with which to render a sound emitter's marker cube in the editor when the emitter's sound is playing but out of the range of the listener. |
| ColorI SFXEmitter::renderColorRangeSphere | [static, inherited]  
The color of the range sphere with which to render sound emitters in the editor. |
| ColorI SFXEmitter::renderColorStoppedInRange | [static, inherited]  
The color of the sound emitter's marker cube in the editor when the emitter's sound is stopped. |
The color with which to render a sound emitter's marker cube in the editor when the emitter's sound is not playing but the emitter is in range of the listener.

**Color** `SFXEmitter::renderColorStoppedOutOfRange` [static, inherited]

The color with which to render a sound emitter's marker cube in the editor when the emitter's sound is not playing and the emitter is out of range of the listener.

**bool** `SFXEmitter::renderEmitters` [static, inherited]

Whether to render enhanced range feedback in the editor on all emitters regardless of selection state.

**float** `SFXEmitter::renderPointDistance` [static, inherited]

The distance between individual points in the sound emitter rendering in the editor as the points move from the emitter's center away to maxDistance.

**float** `SFXEmitter::renderRadialIncrements` [static, inherited]

The stepping (in degrees) for the radial sweep along the axis of the XY plane sweep for sound emitter rendering in the editor.

**float** `SFXEmitter::renderSweepIncrements` [static, inherited]

The stepping (in degrees) for the radial sweep on the XY plane for sound emitter rendering in the editor.
int $SFX::sourceUpdateTime

Milliseconds spent on the last SFXSource update loop.

Sound System Updates

Copyright © GarageGames, LLC. All Rights Reserved.
FMOD
[Sound]

Functionality specific to the FMOD SFX implementation. More...
## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
<th>More...</th>
</tr>
</thead>
<tbody>
<tr>
<td>SFXFMODEvent</td>
<td>A playable sound event in an FMOD Designer audio project.</td>
<td></td>
</tr>
<tr>
<td>SFXFMODEventGroup</td>
<td>A group of events in an imported FMOD Designer project.</td>
<td></td>
</tr>
<tr>
<td>SFXFMODEventSource</td>
<td>A sound source controller playing an FMOD Designer event (SFXFMODEvent).</td>
<td></td>
</tr>
<tr>
<td>SFXFMODProject</td>
<td>An FMOD Designer project loaded into Torque.</td>
<td></td>
</tr>
</tbody>
</table>
## Functions

<table>
<thead>
<tr>
<th>void fmodDumpDSPInfo ()</th>
<th>Dump information about the standard DSP effects.</th>
</tr>
</thead>
<tbody>
<tr>
<td>void fmodDumpMemoryStats ()</td>
<td></td>
</tr>
</tbody>
</table>
### Variables

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>$pref::SFX::FMOD::disableSoftware</td>
<td>Whether to disable the FMOD software mixer to conserve memory.</td>
</tr>
<tr>
<td>string</td>
<td>$pref::SFX::FMOD::DSoundHRTF</td>
<td>The type of HRTF to use for hardware-mixed 3D sounds when FMOD is using DirectSound for sound output and hardware-acceleration is not available.</td>
</tr>
<tr>
<td>bool</td>
<td>$pref::SFX::FMOD::enableProfile</td>
<td>Whether to enable support for FMOD's profiler.</td>
</tr>
<tr>
<td>int</td>
<td>$SFX::Device::fmodCoreMem</td>
<td>Current number of bytes allocated by the core FMOD sound system.</td>
</tr>
<tr>
<td>int</td>
<td>$SFX::Device::fmodEventMem</td>
<td>Current number of bytes allocated by the FMOD Designer event system.</td>
</tr>
<tr>
<td>int</td>
<td>$SFX::Device::fmodNumEventSources</td>
<td>The current number of SFXFMODEventSource instances in the system.</td>
</tr>
<tr>
<td>string</td>
<td>$pref::SFX::FMOD::pluginPath</td>
<td>Path to additional FMOD plugins.</td>
</tr>
<tr>
<td>bool</td>
<td>$pref::SFX::FMOD::useSoftwareHRTF</td>
<td>Whether to enable HRTF in FMOD's software mixer.</td>
</tr>
<tr>
<td>bool</td>
<td>$pref::SFX::FMOD::useSoftwareReverbLowmem</td>
<td>If true, FMOD's SFX reverb is run using 22/24kHz delay buffers, halving the memory required.</td>
</tr>
</tbody>
</table>
Detailed Description

Functionality specific to the FMOD SFX implementation.

When using FMOD for audio output in combination with Torque's sound system, an extended set of features is available to the user. This includes:

- Reverb support
- Enhanced voice virtualization
- Support for multiple listeners
- Enhanced sound format support
  - .aiff
  - .asf
  - .asx
  - .dls
  - .flac
  - .fsb
  - .it
  - .m3u
  - .mid
  - .mod
  - .mp2
  - .mp3
  - .ogg
  - .pls
  - .s3m
  - .vag
  - .wav
  - .wax
  - .wma
  - .xm
  - .xma (on Xbox only)
- FMOD Designer enhanced audio design support

See also:

http://www.fmod.org
FMOD Designer Audio

Using the FMOD Profiler with Torque

FMOD's profiler tool allows to inspect the operation of the FMOD sound system at runtime. To use the tool with Torque, set $pref::SFX::FMOD::enableProfile to true and either restart Torque or reinitialize the FMOD device (e.g. by switching back and forth to the device). After that, you can connect to the running Torque instance from within the FMOD profiler tool.
Function Documentation

void fmodDumpDSPInfo( )

Dump information about the standard DSP effects.

void fmodDumpMemoryStats( )

Returns:
Prints the current memory consumption of the FMOD module
Variable Documentation

bool $pref::SFX::FMOD::disableSoftware

Whether to disable the FMOD software mixer to conserve memory.

All sounds not created with SFXDescription::useHardware or using DSP effects will fail to load.

Note:
Only applies when using an FMOD sound device.

string $pref::SFX::FMOD::DSoundHRTF

The type of HRTF to use for hardware-mixed 3D sounds when FMOD is using DirectSound for sound output and hardware-acceleration is not available.

Options are

- "none": simple stereo panning/doppler/attenuation
- "light": slightly higher quality than "none"
- "full": full quality 3D playback

Note:
Only applies when using an FMOD sound device.

bool $pref::SFX::FMOD::enableProfile

Whether to enable support for FMOD's profiler.

Note:
Only applies when using an FMOD sound device.
Using the FMOD Profiler with Torque

**int $SFX::Device::fmodCoreMem**

Current number of bytes allocated by the core FMOD sound system.

**Note:**
Only relevant if an FMOD sound device is used.

**int $SFX::Device::fmodEventMem**

Current number of bytes allocated by the FMOD Designer event system.

**Note:**
Only relevant if an FMOD sound device is used and the FMOD event DLL is loaded.

**int $SFX::Device::fmodNumEventSources**

The current number of **SFXFMODEventSource** instances in the system.

This tells the number of sounds in the system that are currently playing FMOD Designer events.

**Note:**
Only relevant if an FMOD sound device is used.

**string $pref::SFX::FMOD::pluginPath**

Path to additional FMOD plugins.
**Note:**
Only applies when using an FMOD sound device.

```c
bool $pref::SFX::FMOD::useSoftwareHRTF
```

Whether to enable HRTF in FMOD's software mixer.

This will add a lowpass filter effect to the DSP effect chain of all sounds mixed in software.

**Note:**
Only applies when using an FMOD sound device.

```c
bool $pref::SFX::FMOD::useSoftwareReverbLowmem
```

If true, FMOD's SFX reverb is run using 22/24kHz delay buffers, halving the memory required.

**Note:**
Only applies when using an FMOD sound device.
UNDOCUMENTED
## Functions

<table>
<thead>
<tr>
<th>void</th>
<th>deleteDataBlocks ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Delete all the datablocks we've downloaded.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>preloadClientDataBlocks ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Preload all datablocks in client mode.</td>
</tr>
</tbody>
</table>
### Variables

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><code>aiAddPlayer</code></td>
<td>aiAddPlayer( 'playerName'[,'AI ClassType'] );</td>
</tr>
<tr>
<td>void</td>
<td><code>cancel</code></td>
<td>cancel(eventId)</td>
</tr>
<tr>
<td>void</td>
<td><code>cancelAll</code></td>
<td>cancelAll(objectId): cancel pending events on the specified object. Events will be automatically cancelled if object is deleted.</td>
</tr>
<tr>
<td>void</td>
<td><code>cancelServerQuery</code></td>
<td>cancelServerQuery()</td>
</tr>
<tr>
<td>void</td>
<td><code>clearClientPaths</code></td>
<td>clearClientPaths</td>
</tr>
<tr>
<td>void</td>
<td><code>clearServerPaths</code></td>
<td>clearServerPaths</td>
</tr>
<tr>
<td>void</td>
<td><code>enableWinConsole</code></td>
<td>enableWinConsole(bool);</td>
</tr>
<tr>
<td>string</td>
<td><code>eval</code></td>
<td>eval(consoleString)</td>
</tr>
<tr>
<td>int</td>
<td><code>getEventTimeLeft</code></td>
<td>getEventTimeLeft(scheduleId) Get the time left in ms until this event will trigger.</td>
</tr>
<tr>
<td>string</td>
<td><code>getJoystickAxes</code></td>
<td>getJoystickAxes( instance )</td>
</tr>
<tr>
<td>int</td>
<td><code>getMaxFrameAllocation</code></td>
<td>getMaxFrameAllocation();</td>
</tr>
<tr>
<td>int</td>
<td><code>getScheduleDuration</code></td>
<td>getScheduleDuration(scheduleId);</td>
</tr>
<tr>
<td>int</td>
<td><code>getServerCount</code></td>
<td>getServerCount</td>
</tr>
</tbody>
</table>
```plaintext
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>getServerCount()</code></td>
<td></td>
</tr>
<tr>
<td><code>int getTimeSinceStart</code></td>
<td></td>
</tr>
<tr>
<td><code>getTimeSinceStart(scheduleId)</code></td>
<td></td>
</tr>
<tr>
<td><code>string getUserDataDirectory()</code></td>
<td></td>
</tr>
<tr>
<td><code>getUserDataDirectory()</code></td>
<td></td>
</tr>
<tr>
<td><code>string getUserHomeDirectory()</code></td>
<td></td>
</tr>
<tr>
<td><code>getUserHomeDirectory()</code></td>
<td></td>
</tr>
<tr>
<td><code>bool isDemo</code></td>
<td></td>
</tr>
<tr>
<td><code>bool isEventPending</code></td>
<td></td>
</tr>
<tr>
<td><code>isEventPending(scheduleId)</code></td>
<td></td>
</tr>
<tr>
<td><code>bool isJoystickDetected</code></td>
<td></td>
</tr>
<tr>
<td><code>isJoystickDetected()</code></td>
<td></td>
</tr>
<tr>
<td><code>bool isKoreanBuild</code></td>
<td></td>
</tr>
<tr>
<td><code>isKoreanBuild()</code></td>
<td></td>
</tr>
<tr>
<td><code>bool isObject</code></td>
<td></td>
</tr>
<tr>
<td><code>isObject(object)</code></td>
<td></td>
</tr>
<tr>
<td><code>static bool TerrainBlock::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool InteriorInstance::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool ForestWindEmitter::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool Forest::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool TimeOfDay::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool Sun::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool SkyBox::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
</tbody>
</table>
```
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>static bool ScatterSky::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool River::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool MeshRoad::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool DecalRoad::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool WheeledVehicle::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool FlyingVehicle::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool AITurretShape::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool TurretShape::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool SFXEmitter::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool PxMultiActor::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool PhysicsShape::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool PhysicsForce::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool Precipitation::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool ParticleEmitterNode::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool Lightning::isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>Class/Type</td>
<td>Method</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>RenderMeshExample</td>
<td>isRenderable</td>
</tr>
<tr>
<td>TSStatic</td>
<td>isRenderable</td>
</tr>
<tr>
<td>Trigger</td>
<td>isRenderable</td>
</tr>
<tr>
<td>RigidShape</td>
<td>isRenderable</td>
</tr>
<tr>
<td>ProximityMine</td>
<td>isRenderable</td>
</tr>
<tr>
<td>Projectile</td>
<td>isRenderable</td>
</tr>
<tr>
<td>Prefab</td>
<td>isRenderable</td>
</tr>
<tr>
<td>Portal</td>
<td>isRenderable</td>
</tr>
<tr>
<td>Zone</td>
<td>isRenderable</td>
</tr>
<tr>
<td>PhysicalZone</td>
<td>isRenderable</td>
</tr>
<tr>
<td>PathCamera</td>
<td>isRenderable</td>
</tr>
<tr>
<td>SpawnSphere</td>
<td>isRenderable</td>
</tr>
<tr>
<td>Item</td>
<td>isRenderable</td>
</tr>
<tr>
<td>GroundPlane</td>
<td>isRenderable</td>
</tr>
<tr>
<td>Debris</td>
<td>isRenderable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static bool</th>
<th>Class Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera::isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td>AIPlayer::isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td>Player::isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td>Marker::isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td>WaterPlane::isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td>WaterBlock::isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td>WaterObject::isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td>CloudLayer::isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td>BasicClouds::isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td>HoverVehicle::isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td>Vehicle::isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td>SFXSpace::isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td>PxCloth::isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td>PhysicsDebris::isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td>Splash::isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td>Class</td>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>--------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>ParticleEmitter</td>
<td>static bool isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>GroundCover</td>
<td>static bool isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>fxShapeReplicatedStatic</td>
<td>static bool isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>fxShapeReplicator</td>
<td>static bool isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>fxFoliageReplicator</td>
<td>static bool isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>Explosion</td>
<td>static bool isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>RenderShapeExample</td>
<td>static bool isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>RenderObjectExample</td>
<td>static bool isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>DecalManager</td>
<td>static bool isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>SpotLight</td>
<td>static bool isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>StaticShape</td>
<td>static bool isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>PointLight</td>
<td>static bool isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>LightBase</td>
<td>static bool isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>OcclusionVolume</td>
<td>static bool isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>WayPoint</td>
<td>static bool isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Static bool</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>MissionMarker::isRenderable</code></td>
<td>Enables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td><code>ShapeBase::isRenderable</code></td>
<td>Enables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td><code>GameBase::isRenderable</code></td>
<td>Enables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td><code>ConvexShape::isRenderable</code></td>
<td>Enables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td><code>SceneObject::isRenderable</code></td>
<td>Enables rendering of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td><code>TerrainBlock::isSelectable</code></td>
<td>Enables selection of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td><code>InteriorInstance::isSelectable</code></td>
<td>Enables selection of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td><code>ForestWindEmitter::isSelectable</code></td>
<td>Enables selection of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td><code>Forest::isSelectable</code></td>
<td>Enables selection of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td><code>TimeOfDay::isSelectable</code></td>
<td>Enables selection of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td><code>Sun::isSelectable</code></td>
<td>Enables selection of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td><code>SkyBox::isSelectable</code></td>
<td>Enables selection of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td><code>ScatterSky::isSelectable</code></td>
<td>Enables selection of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td><code>River::isSelectable</code></td>
<td>Enables selection of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td><code>MeshRoad::isSelectable</code></td>
<td>Enables selection of all instances of this type.</td>
<td></td>
</tr>
<tr>
<td>Class</td>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>DecalRoad</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>WheeledVehicle</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>FlyingVehicle</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>AITurretShape</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>TurretShape</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>SFXEmitter</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>PxMultiActor</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>PhysicsShape</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>PhysicsForce</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>Precipitation</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>ParticleEmitterNode</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>Lightning</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>RenderMeshExample</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>TSStatic</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>Trigger</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
static bool **RigidShape::isSelectable**
Disables selection of all instances of this type.

static bool **ProximityMine::isSelectable**
Disables selection of all instances of this type.

static bool **Projectile::isSelectable**
Disables selection of all instances of this type.

static bool **Prefab::isSelectable**
Disables selection of all instances of this type.

static bool **Portal::isSelectable**
Disables selection of all instances of this type.

static bool **Zone::isSelectable**
Disables selection of all instances of this type.

static bool **PhysicalZone::isSelectable**
Disables selection of all instances of this type.

static bool **PathCamera::isSelectable**
Disables selection of all instances of this type.

static bool **SpawnSphere::isSelectable**
Disables selection of all instances of this type.

static bool **Item::isSelectable**
Disables selection of all instances of this type.

static bool **GroundPlane::isSelectable**
Disables selection of all instances of this type.

static bool **Debris::isSelectable**
Disables selection of all instances of this type.

static bool **Camera::isSelectable**
Disables selection of all instances of this type.

static bool **AIPlayer::isSelectable**
Disables selection of all instances of this type.

static bool **Player::isSelectable**
Disables selection of all instances of this type.
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Marker::isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td><code>WaterPlane::isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td><code>WaterBlock::isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td><code>WaterObject::isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td><code>CloudLayer::isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td><code>BasicClouds::isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td><code>HoverVehicle::isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td><code>Vehicle::isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td><code>SFXSpace::isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td><code>PxCloth::isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td><code>PhysicsDebris::isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td><code>Splash::isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td><code>ParticleEmitter::isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td><code>GroundCover::isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td><code>fxShapeReplicatedStatic::isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>

The table lists various static boolean methods associated with different types, each disabling the selection of all instances of that type.
static bool fxShapeReplicator::isSelectable
Disables selection of all instances of this type.

static bool fxFoliageReplicator::isSelectable
Disables selection of all instances of this type.

static bool Explosion::isSelectable
Disables selection of all instances of this type.

static bool RenderShapeExample::isSelectable
Disables selection of all instances of this type.

static bool RenderObjectExample::isSelectiveable
Disables selection of all instances of this type.

static bool DecalManager::isSelectable
Disables selection of all instances of this type.

static bool SpotLight::isSelectable
Disables selection of all instances of this type.

static bool StaticShape::isSelectable
Disables selection of all instances of this type.

static bool PointLight::isSelectable
Disables selection of all instances of this type.

static bool LightBase::isSelectiveable
Disables selection of all instances of this type.

static bool OcclusionVolume::isSelectiveable
Disables selection of all instances of this type.

static bool WayPoint::isSelectiveable
Disables selection of all instances of this type.

static bool MissionMarker::isSelectiveable
Disables selection of all instances of this type.

static bool ShapeBase::isSelectiveable
Disables selection of all instances of this type.

static bool GameBase::isSelectiveable
Disables selection of all instances of this type.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>static bool ConvexShape::isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool SceneObject::isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td><code>bool isWebDemo</code></td>
<td></td>
</tr>
<tr>
<td><code>float $pref::PhysicsDebris::lifetimeScale</code></td>
<td>Scales how long PhysicsDebris will live before being removed.</td>
</tr>
<tr>
<td><code>int nameToID</code></td>
<td>nameToID(object)</td>
</tr>
<tr>
<td><code>static bool PhysicsShape::noCorrections</code></td>
<td>Determines if the shape will receive corrections from the server or will instead be allowed to diverge.</td>
</tr>
<tr>
<td><code>static bool PhysicsShape::noSmoothing</code></td>
<td>Determines if client-side shapes will attempt to smoothly transition to their new position after receiving a correction.</td>
</tr>
<tr>
<td><code>void physicsDebugDraw</code></td>
<td>physicsDebugDraw( bool enable )</td>
</tr>
<tr>
<td><code>void physicsDestroy</code></td>
<td>physicsDestroy()</td>
</tr>
<tr>
<td><code>void physicsDestroyWorld</code></td>
<td>physicsDestroyWorld( String worldName )</td>
</tr>
<tr>
<td><code>float physicsGetTimeScale</code></td>
<td>physicsGetTimeScale()</td>
</tr>
<tr>
<td><code>bool physicsInit</code></td>
<td>physicsInit( [string library] )</td>
</tr>
<tr>
<td><code>bool physicsInitWorld</code></td>
<td>physicsInitWorld( String worldName )</td>
</tr>
<tr>
<td><code>void physicsRestoreState</code></td>
<td>physicsRestoreState()</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td><code>void physicsSetTimeScale</code></td>
<td>Set the physics time scale.</td>
</tr>
<tr>
<td><code>physicsSetTimeScale(float scale)</code></td>
<td>Set the physics time scale.</td>
</tr>
<tr>
<td><code>bool physicsSimulationEnabled</code></td>
<td>Check if simulation is enabled.</td>
</tr>
<tr>
<td><code>physicsSimulationEnabled()</code></td>
<td>Check if simulation is enabled.</td>
</tr>
<tr>
<td><code>void physicsStartSimulation</code></td>
<td>Start the physics simulation.</td>
</tr>
<tr>
<td><code>physicsStartSimulation(string worldName)</code></td>
<td>Start the physics simulation.</td>
</tr>
<tr>
<td><code>void physicsStopSimulation</code></td>
<td>Stop the physics simulation.</td>
</tr>
<tr>
<td><code>physicsStopSimulation(string worldName)</code></td>
<td>Stop the physics simulation.</td>
</tr>
<tr>
<td><code>void physicsStoreState</code></td>
<td>Store the physics state.</td>
</tr>
<tr>
<td><code>physicsStoreState()</code></td>
<td>Store the physics state.</td>
</tr>
<tr>
<td><code>bool physXRemoteDebuggerConnect</code></td>
<td>Connect to physX remote debugger.</td>
</tr>
<tr>
<td><code>void physXRemoteDebuggerDisconnect</code></td>
<td>Disconnect from physX remote debugger.</td>
</tr>
<tr>
<td><code>void queryAllServers</code></td>
<td>Query all servers.</td>
</tr>
<tr>
<td><code>queryAllServers(...)</code></td>
<td>Query all servers.</td>
</tr>
<tr>
<td><code>void queryLanServers</code></td>
<td>Query LAN servers.</td>
</tr>
<tr>
<td><code>queryLanServers(...)</code></td>
<td>Query LAN servers.</td>
</tr>
<tr>
<td><code>void queryMasterServer</code></td>
<td>Query master server.</td>
</tr>
<tr>
<td><code>queryMasterServer(...)</code></td>
<td>Query master server.</td>
</tr>
<tr>
<td><code>void querySingleServer</code></td>
<td>Query single server.</td>
</tr>
<tr>
<td><code>querySingleServer(address, flags)</code></td>
<td>Query single server.</td>
</tr>
<tr>
<td><code>void restartInstance</code></td>
<td>Restart the instance.</td>
</tr>
<tr>
<td><code>restartInstance()</code></td>
<td>Restart the instance.</td>
</tr>
<tr>
<td><code>void sbmDumpStats</code></td>
<td>Dump stats.</td>
</tr>
<tr>
<td><code>void sbmDumpStrings</code></td>
<td>Dump strings.</td>
</tr>
<tr>
<td><code>int schedule</code></td>
<td>Schedule task.</td>
</tr>
<tr>
<td>`schedule(time, refobject</td>
<td>0, command, &lt;arg1...argN&gt;)`</td>
</tr>
<tr>
<td><code>bool setServerInfo</code></td>
<td>Set server info.</td>
</tr>
<tr>
<td><code>setServerInfo(index)</code></td>
<td>Set server info.</td>
</tr>
<tr>
<td><code>bool setShadowManager</code></td>
<td>Set shadow manager.</td>
</tr>
<tr>
<td><code>string sShadowSystemName</code></td>
<td>Get shadow system name.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>string setShadowVizLight</td>
<td></td>
</tr>
<tr>
<td>void startHeartbeat</td>
<td></td>
</tr>
<tr>
<td>void startHeartbeat()</td>
<td></td>
</tr>
<tr>
<td>int startPrecisionTimer</td>
<td>- Create and start a high resolution platform timer. Returns the timer id.</td>
</tr>
<tr>
<td>int startPrecisionTimer()</td>
<td></td>
</tr>
<tr>
<td>void stopHeartbeat</td>
<td></td>
</tr>
<tr>
<td>void stopHeartbeat()</td>
<td></td>
</tr>
<tr>
<td>int stopPrecisionTimer</td>
<td>- Stop and destroy timer with the passed id. Returns the elapsed milliseconds.</td>
</tr>
<tr>
<td>int stopPrecisionTimer( S32 id )</td>
<td></td>
</tr>
<tr>
<td>void stopServerQuery</td>
<td></td>
</tr>
<tr>
<td>void stopServerQuery()</td>
<td></td>
</tr>
<tr>
<td>string strToPlayerName</td>
<td></td>
</tr>
<tr>
<td>string strToPlayerName( string )</td>
<td></td>
</tr>
<tr>
<td>string TestFunction2Args</td>
<td></td>
</tr>
<tr>
<td>string TestFunction2Args( arg1, arg2 )</td>
<td></td>
</tr>
<tr>
<td>string testJavaScriptBridge</td>
<td></td>
</tr>
<tr>
<td>string testJavaScriptBridge( arg1, arg2, arg3 )</td>
<td></td>
</tr>
<tr>
<td>void tsUpdateImposterImages</td>
<td></td>
</tr>
<tr>
<td>void tsUpdateImposterImages( bool forceupdate )</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description
Function Documentation

void deleteDataBlocks()

Delete all the datablocks we've downloaded.

This is usually done in preparation of downloading a new set of datablocks, such as occurs on a mission change, but it's also good post-mission cleanup.

void preloadClientDataBlocks()

Preload all datablocks in client mode.

(Server parameter is set to false). This will take some time to complete.
Variable Documentation

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int aiAddPlayer</td>
<td></td>
</tr>
<tr>
<td>aiAddPlayer( 'playerName'[,'AIClassType'] );</td>
<td></td>
</tr>
<tr>
<td>void cancel</td>
<td></td>
</tr>
<tr>
<td>cancel(eventId)</td>
<td>cancelAll(objectId): cancel pending events on the specified object. Events will be automatically cancelled if object is deleted.</td>
</tr>
<tr>
<td>void cancelAll</td>
<td></td>
</tr>
<tr>
<td>clearClientPaths</td>
<td>UNDOCUMENTED!</td>
</tr>
<tr>
<td>clearServerPaths</td>
<td>UNDOCUMENTED!</td>
</tr>
<tr>
<td>dumpProcessList</td>
<td></td>
</tr>
</tbody>
</table>

```cpp
void enableWinConsole

enableWinConsole(bool);

string eval

eval(consoleString)

int getEventTimeLeft

gGetEventTimeLeft(scheduleId) Get the time left in ms until this event will trigger.

string getJoystickAxes

gGetJoystickAxes(instance)

int getMaxFrameAllocation

getMaxFrameAllocation();

int getScheduleDuration

gGetScheduleDuration(scheduleId);
```
int getServerCount

getServerCount();

int getTimeSinceStart

getTimeSinceStart(scheduleId);

string getUserDataDirectory

getUserDataDirectory();

string getUserHomeDirectory

getUserHomeDirectory();

bool isDemo

UNDOCUMENTED!

bool isEventPending

isEventPending(scheduleId);

bool isJoystickDetected

isJoystickDetected();

bool isKoreanBuild
<table>
<thead>
<tr>
<th>Function/Method</th>
<th>Description</th>
<th>Inheritance</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>isKoreanBuild()</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>bool isObject</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>isObject(object)</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>bool TerrainBlock::isRenderable [static, inherited]</code></td>
<td>Disables rendering of all instances of this type.</td>
<td>Reimplemented from SceneObject.</td>
</tr>
<tr>
<td><code>bool InteriorInstance::isRenderable [static, inherited]</code></td>
<td>Disables rendering of all instances of this type.</td>
<td>Reimplemented from SceneObject.</td>
</tr>
<tr>
<td><code>bool ForestWindEmitter::isRenderable [static, inherited]</code></td>
<td>Disables rendering of all instances of this type.</td>
<td>Reimplemented from SceneObject.</td>
</tr>
<tr>
<td><code>bool Forest::isRenderable [static, inherited]</code></td>
<td>Disables rendering of all instances of this type.</td>
<td>Reimplemented from SceneObject.</td>
</tr>
</tbody>
</table>
bool TimeOfDay::isRenderable [static, inherited]

Disables rendering of all instances of this type.

Reimplemented from SceneObject.

bool Sun::isRenderable [static, inherited]

Disables rendering of all instances of this type.

Reimplemented from SceneObject.

bool SkyBox::isRenderable [static, inherited]

Disables rendering of all instances of this type.

Reimplemented from SceneObject.

bool ScatterSky::isRenderable [static, inherited]

Disables rendering of all instances of this type.

Reimplemented from SceneObject.

bool River::isRenderable [static, inherited]

Disables rendering of all instances of this type.

Reimplemented from WaterObject.

bool MeshRoad::isRenderable [static, inherited]
Disables rendering of all instances of this type.
Reimplemented from SceneObject.

`bool DecalRoad::isRenderable [static, inherited]`
Disables rendering of all instances of this type.
Reimplemented from SceneObject.

`bool WheeledVehicle::isRenderable [static, inherited]`
Disables rendering of all instances of this type.
Reimplemented from Vehicle.

`bool FlyingVehicle::isRenderable [static, inherited]`
Disables rendering of all instances of this type.
Reimplemented from Vehicle.

`bool AI turretShape::isRenderable [static, inherited]`
Disables rendering of all instances of this type.
Reimplemented from TurretShape.

`bool TurretShape::isRenderable [static, inherited]`
Disables rendering of all instances of this type.
Reimplemented from Item.

Reimplemented in ALTurretShape.

```cpp
bool SFXEmitter::isRenderable [static, inherited]
```

Disables rendering of all instances of this type.

Reimplemented from SceneObject.

```cpp
bool PxMultiActor::isRenderable [static, inherited]
```

Disables rendering of all instances of this type.

Reimplemented from GameBase.

```cpp
bool PhysicsShape::isRenderable [static, inherited]
```

Disables rendering of all instances of this type.

Reimplemented from GameBase.

```cpp
bool PhysicsForce::isRenderable [static, inherited]
```

Disables rendering of all instances of this type.

Reimplemented from SceneObject.

```cpp
bool Precipitation::isRenderable [static, inherited]
```

Disables rendering of all instances of this type.
Reimplemented from GameBase.

bool ParticleEmitterNode::isRenderable [static, inherited]

Disables rendering of all instances of this type.
Reimplemented from GameBase.

bool Lightning::isRenderable [static, inherited]

Disables rendering of all instances of this type.
Reimplemented from GameBase.

bool RenderMeshExample::isRenderable [static, inherited]

Disables rendering of all instances of this type.
Reimplemented from SceneObject.

bool TSStatic::isRenderable [static, inherited]

Disables rendering of all instances of this type.
Reimplemented from SceneObject.

bool Trigger::isRenderable [static, inherited]

Disables rendering of all instances of this type.
Reimplemented from GameBase.
<table>
<thead>
<tr>
<th>Class</th>
<th>Method</th>
<th>Inheritance</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RigidShape</td>
<td>isRenderable</td>
<td>[static, inherited]</td>
<td>Disables rendering of all instances of this type. Reimplemented from ShapeBase.</td>
</tr>
<tr>
<td>ProximityMine</td>
<td>isRenderable</td>
<td>[static, inherited]</td>
<td>Disables rendering of all instances of this type. Reimplemented from Item.</td>
</tr>
<tr>
<td>Projectile</td>
<td>isRenderable</td>
<td>[static, inherited]</td>
<td>Disables rendering of all instances of this type. Reimplemented from GameBase.</td>
</tr>
<tr>
<td>Prefab</td>
<td>isRenderable</td>
<td>[static, inherited]</td>
<td>Disables rendering of all instances of this type. Reimplemented from SceneObject.</td>
</tr>
<tr>
<td>Portal</td>
<td>isRenderable</td>
<td>[static, inherited]</td>
<td>Disables rendering of all instances of this type. Reimplemented from Zone.</td>
</tr>
<tr>
<td>Zone</td>
<td>isRenderable</td>
<td>[static, inherited]</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
</tbody>
</table>
Disables rendering of all instances of this type.

Reimplemented from SceneObject.

Reimplemented in Portal.

bool PhysicalZone::isRenderable [static, inherited]

Disables rendering of all instances of this type.

Reimplemented from SceneObject.

bool PathCamera::isRenderable [static, inherited]

Disables rendering of all instances of this type.

Reimplemented from ShapeBase.

bool SpawnSphere::isRenderable [static, inherited]

Disables rendering of all instances of this type.

Reimplemented from MissionMarker.

bool Item::isRenderable [static, inherited]

Disables rendering of all instances of this type.

Reimplemented from ShapeBase.

Reimplemented in ProximityMine, TurretShape, and AI:TurretShape.
**bool GroundPlane::isRenderable** [static, inherited]

Disables rendering of all instances of this type.
Reimplemented from *SceneObject*.

**bool Debris::isRenderable** [static, inherited]

Disables rendering of all instances of this type.
Reimplemented from *GameBase*.

**bool Camera::isRenderable** [static, inherited]

Disables rendering of all instances of this type.
Reimplemented from *ShapeBase*.

**bool AIPlayer::isRenderable** [static, inherited]

Disables rendering of all instances of this type.
Reimplemented from *Player*.

**bool Player::isRenderable** [static, inherited]

Disables rendering of all instances of this type.
Reimplemented from *ShapeBase*.
Reimplemented in *AIPlayer*. 
bool Marker::isRenderable [static, inherited]

Disables rendering of all instances of this type.

Reimplemented from SceneObject.

bool WaterPlane::isRenderable [static, inherited]

Disables rendering of all instances of this type.

Reimplemented from WaterObject.

bool WaterBlock::isRenderable [static, inherited]

Disables rendering of all instances of this type.

Reimplemented from WaterObject.

bool WaterObject::isRenderable [static, inherited]

Disables rendering of all instances of this type.

Reimplemented from SceneObject.

Reimplemented in WaterBlock, WaterPlane, and River.

bool CloudLayer::isRenderable [static, inherited]

Disables rendering of all instances of this type.

Reimplemented from SceneObject.
<table>
<thead>
<tr>
<th>bool BasicClouds::isRenderable [static, inherited]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>Reimplemented from SceneObject.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool HoverVehicle::isRenderable [static, inherited]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>Reimplemented from Vehicle.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool Vehicle::isRenderable [static, inherited]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>Reimplemented from ShapeBase.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool SFXSpace::isRenderable [static, inherited]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>Reimplemented from SceneObject.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool PxCloth::isRenderable [static, inherited]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>Reimplemented from GameBase.</td>
</tr>
</tbody>
</table>
bool PhysicsDebris::isRenderable [static, inherited]

Disables rendering of all instances of this type.
Reimplemented from GameBase.

bool Splash::isRenderable [static, inherited]

Disables rendering of all instances of this type.
Reimplemented from GameBase.

bool ParticleEmitter::isRenderable [static, inherited]

Disables rendering of all instances of this type.
Reimplemented from GameBase.

bool GroundCover::isRenderable [static, inherited]

Disables rendering of all instances of this type.
Reimplemented from SceneObject.

bool fxShapeReplicatedStatic::isRenderable [static, inherited]

Disables rendering of all instances of this type.
Reimplemented from SceneObject.

bool fxShapeReplicator::isRenderable [static, inherited]
Disables rendering of all instances of this type.
Reimplemented from SceneObject.

bool fxFoliageReplicator::isRenderable [static, inherited]
Disables rendering of all instances of this type.
Reimplemented from SceneObject.

bool Explosion::isRenderable [static, inherited]
Disables rendering of all instances of this type.
Reimplemented from GameBase.

bool RenderShapeExample::isRenderable [static, inherited]
Disables rendering of all instances of this type.
Reimplemented from SceneObject.

bool RenderObjectExample::isRenderable [static, inherited]
Disables rendering of all instances of this type.
Reimplemented from SceneObject.

bool DecalManager::isRenderable [static, inherited]
Disables rendering of all instances of this type.
Reimplemented from `SceneObject`.

```cpp
bool SpotLight::isRenderable [static, inherited]
```

Disables rendering of all instances of this type.

Reimplemented from `LightBase`.

```cpp
bool StaticShape::isRenderable [static, inherited]
```

Disables rendering of all instances of this type.

Reimplemented from `ShapeBase`.

```cpp
bool PointLight::isRenderable [static, inherited]
```

Disables rendering of all instances of this type.

Reimplemented from `LightBase`.

```cpp
bool LightBase::isRenderable [static, inherited]
```

Disables rendering of all instances of this type.

Reimplemented from `SceneObject`.

Reimplemented in `PointLight`, and `SpotLight`.

```cpp
bool OcclusionVolume::isRenderable [static, inherited]
```

Disables rendering of all instances of this type.
Reimplemented from *SceneObject.*

```cpp
bool WayPoint::isRenderable [static, inherited]
```

Disables rendering of all instances of this type.

Reimplemented from *MissionMarker.*

```cpp
bool MissionMarker::isRenderable [static, inherited]
```

Disables rendering of all instances of this type.

Reimplemented from *ShapeBase.*

Reimplemented in *WayPoint,* and *SpawnSphere.*

```cpp
bool ShapeBase::isRenderable [static, inherited]
```

Disables rendering of all instances of this type.

Reimplemented from *GameBase.*

Reimplemented in *MissionMarker,* *WayPoint,* *StaticShape,* *Vehicle,* *HoverVehicle,* *Player,* *AIPlayer,* *Camera,* *Item,* *SpawnSphere,* *PathCamera,* *ProximityMine,* *RigidShape,* *TurretShape,* *AITurretShape,* *FlyingVehicle,* and *WheeledVehicle.*

```cpp
bool GameBase::isRenderable [static, inherited]
```

Disables rendering of all instances of this type.

Reimplemented from *SceneObject.*

Reimplemented in *ShapeBase,* *MissionMarker,* *WayPoint,*
StaticShape, Explosion, ParticleEmitter, Splash, PhysicsDebris, PxCloth, Vehicle, HoverVehicle, Player, AIPlayer, Camera, Debris, Item, SpawnSphere, PathCamera, Projectile, ProximityMine, RigidShape, Trigger, Lightning, ParticleEmitterNode, Precipitation, PhysicsShape, PxMultiActor, TurretShape, AITurretShape, FlyingVehicle, and WheeledVehicle.

<table>
<thead>
<tr>
<th>bool ConvexShape::isRenderable [static, inherited]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>Reimplemented from SceneObject.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool SceneObject::isRenderable [static, inherited]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>Reimplemented in ConvexShape, GameBase, ShapeBase,</td>
</tr>
<tr>
<td>MissionMarker, WayPoint, OcclusionVolume, LightBase,</td>
</tr>
<tr>
<td>PointLight, StaticShape, SpotLight, DecalManager,</td>
</tr>
<tr>
<td>RenderObjectExample, RenderShapeExample, Explosion,</td>
</tr>
<tr>
<td>fxFoliageReplicator, fxShapeReplicator, fxShapeReplicatedStatic,</td>
</tr>
<tr>
<td>GroundCover, ParticleEmitter, Splash, PhysicsDebris, PxCloth,</td>
</tr>
<tr>
<td>SFXSpace, Vehicle, HoverVehicle, BasicClouds, CloudLayer,</td>
</tr>
<tr>
<td>WaterObject, WaterBlock, WaterPlane, Marker, Player, AIPlayer,</td>
</tr>
<tr>
<td>Camera, Debris, GroundPlane, Item, SpawnSphere, PathCamera,</td>
</tr>
<tr>
<td>PhysicalZone, Zone, Portal, Prefab, Projectile, ProximityMine,</td>
</tr>
<tr>
<td>RigidShape, Trigger, TSStatic, RenderMeshExample, Lightning,</td>
</tr>
<tr>
<td>ParticleEmitterNode, Precipitation, PhysicsForce, PhysicsShape,</td>
</tr>
<tr>
<td>PxMultiActor, SFXEmitter, TurretShape, AITurretShape,</td>
</tr>
<tr>
<td>FlyingVehicle, WheeledVehicle, DecalRoad, MeshRoad, River,</td>
</tr>
<tr>
<td>ScatterSky, SkyBox, Sun, TimeOfDay, Forest, ForestWindEmitter,</td>
</tr>
<tr>
<td>InteriorInstance, and TerrainBlock.</td>
</tr>
</tbody>
</table>

| bool TerrainBlock::isSelectable [static, inherited] |
Disables selection of all instances of this type.
Reimplemented from SceneObject.

**bool InteriorInstance::isSelectable** [static, inherited]

Disables selection of all instances of this type.
Reimplemented from SceneObject.

**bool ForestWindEmitter::isSelectable** [static, inherited]

Disables selection of all instances of this type.
Reimplemented from SceneObject.

**bool Forest::isSelectable** [static, inherited]

Disables selection of all instances of this type.
Reimplemented from SceneObject.

**bool TimeOfDay::isSelectable** [static, inherited]

Disables selection of all instances of this type.
Reimplemented from SceneObject.

**bool Sun::isSelectable** [static, inherited]

Disables selection of all instances of this type.
<table>
<thead>
<tr>
<th>Class</th>
<th>Method Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SkyBox</td>
<td><code>isSelectable</code></td>
<td>Disables selection of all instances of this type. Reimplemented from SceneObject.</td>
</tr>
<tr>
<td>ScatterSky</td>
<td><code>isSelectable</code></td>
<td>Disables selection of all instances of this type. Reimplemented from SceneObject.</td>
</tr>
<tr>
<td>River</td>
<td><code>isSelectable</code></td>
<td>Disables selection of all instances of this type. Reimplemented from WaterObject.</td>
</tr>
<tr>
<td>MeshRoad</td>
<td><code>isSelectable</code></td>
<td>Disables selection of all instances of this type. Reimplemented from SceneObject.</td>
</tr>
<tr>
<td>DecalRoad</td>
<td><code>isSelectable</code></td>
<td>Disables selection of all instances of this type. Reimplemented from SceneObject.</td>
</tr>
</tbody>
</table>
**bool WheeledVehicle::isSelectable** [static, inherited]

Disables selection of all instances of this type.

Reimplemented from **Vehicle**.

**bool FlyingVehicle::isSelectable** [static, inherited]

Disables selection of all instances of this type.

Reimplemented from **Vehicle**.

**bool AI_TurretShape::isSelectable** [static, inherited]

Disables selection of all instances of this type.

Reimplemented from **TurretShape**.

**bool TurretShape::isSelectable** [static, inherited]

Disables selection of all instances of this type.

Reimplemented from **Item**.

Reimplemented in **AITurretShape**.

**bool SFXEmitter::isSelectable** [static, inherited]

Disables selection of all instances of this type.

Reimplemented from **SceneObject**.
<table>
<thead>
<tr>
<th>Class</th>
<th>Method Name</th>
<th>Description</th>
<th>Inherited From</th>
</tr>
</thead>
<tbody>
<tr>
<td>PxMultiActor</td>
<td>bool isSelectable [static, inherited]</td>
<td>Disables selection of all instances of this type.</td>
<td>GameBase</td>
</tr>
<tr>
<td>PhysicsShape</td>
<td>bool isSelectable [static, inherited]</td>
<td>Disables selection of all instances of this type.</td>
<td>GameBase</td>
</tr>
<tr>
<td>PhysicsForce</td>
<td>bool isSelectable [static, inherited]</td>
<td>Disables selection of all instances of this type.</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Precipitation</td>
<td>bool isSelectable [static, inherited]</td>
<td>Disables selection of all instances of this type.</td>
<td>GameBase</td>
</tr>
<tr>
<td>ParticleEmitterNode</td>
<td>bool isSelectable [static, inherited]</td>
<td>Disables selection of all instances of this type.</td>
<td>GameBase</td>
</tr>
<tr>
<td>Lightning</td>
<td>bool isSelectable [static, inherited]</td>
<td>Disables selection of all instances of this type.</td>
<td>GameBase</td>
</tr>
</tbody>
</table>
Disables selection of all instances of this type.
Reimplemented from GameBase.

bool RenderMeshExample::isSelectable [static, inherited]
Disables selection of all instances of this type.
Reimplemented from SceneObject.

bool TSStatic::isSelectable [static, inherited]
Disables selection of all instances of this type.
Reimplemented from SceneObject.

bool Trigger::isSelectable [static, inherited]
Disables selection of all instances of this type.
Reimplemented from GameBase.

bool RigidShape::isSelectable [static, inherited]
Disables selection of all instances of this type.
Reimplemented from ShapeBase.

bool ProximityMine::isSelectable [static, inherited]
Disables selection of all instances of this type.
Reimplemented from Item.

```cpp
bool Projectile::isSelectable [static, inherited]
```
Disables selection of all instances of this type.
Reimplemented from GameBase.

```cpp
bool Prefab::isSelectable [static, inherited]
```
Disables selection of all instances of this type.
Reimplemented from SceneObject.

```cpp
bool Portal::isSelectable [static, inherited]
```
Disables selection of all instances of this type.
Reimplemented from Zone.

```cpp
bool Zone::isSelectable [static, inherited]
```
Disables selection of all instances of this type.
Reimplemented from SceneObject.
Reimplemented in Portal.

```cpp
bool PhysicalZone::isSelectable [static, inherited]
```
Disables selection of all instances of this type.
Reimplemented from `SceneObject`.

```cpp
bool PathCamera::isSelectable [static, inherited]
```

Disables selection of all instances of this type.

Reimplemented from `ShapeBase`.

```cpp
bool SpawnSphere::isSelectable [static, inherited]
```

Disables selection of all instances of this type.

Reimplemented from `MissionMarker`.

```cpp
bool Item::isSelectable [static, inherited]
```

Disables selection of all instances of this type.

Reimplemented from `ShapeBase`.

Reimplemented in `ProximityMine`, `TurretShape`, and `AI-TurretShape`.

```cpp
bool GroundPlane::isSelectable [static, inherited]
```

Disables selection of all instances of this type.

Reimplemented from `SceneObject`.

```cpp
bool Debris::isSelectable [static, inherited]
```

Disables selection of all instances of this type.
Reimplemented from GameBase.

```cpp
bool Camera::isSelectable [static, inherited]
```
Disables selection of all instances of this type.
Reimplemented from ShapeBase.

```cpp
bool AIPlayer::isSelectable [static, inherited]
```
Disables selection of all instances of this type.
Reimplemented from Player.

```cpp
bool Player::isSelectable [static, inherited]
```
Disables selection of all instances of this type.
Reimplemented from ShapeBase.
Reimplemented in AIPlayer.

```cpp
bool Marker::isSelectable [static, inherited]
```
Disables selection of all instances of this type.
Reimplemented from SceneObject.

```cpp
bool WaterPlane::isSelectable [static, inherited]
```
Disables selection of all instances of this type.
Reimplemented from `WaterObject`.

```cpp
bool WaterBlock::isSelectable [static, inherited]
```

Disables selection of all instances of this type.

Reimplemented from `WaterObject`.

```cpp
bool WaterObject::isSelectable [static, inherited]
```

Disables selection of all instances of this type.

Reimplemented from `SceneObject`.

Reimplemented in `WaterBlock`, `WaterPlane`, and `River`.

```cpp
bool CloudLayer::isSelectable [static, inherited]
```

Disables selection of all instances of this type.

Reimplemented from `SceneObject`.

```cpp
bool BasicClouds::isSelectable [static, inherited]
```

Disables selection of all instances of this type.

Reimplemented from `SceneObject`.

```cpp
bool HoverVehicle::isSelectable [static, inherited]
```

Disables selection of all instances of this type.
Reimplemented from Vehicle.

```cpp
bool Vehicle::isSelectable [static, inherited]
```

Disables selection of all instances of this type.

Reimplemented from ShapeBase.


```cpp
bool SFXSpace::isSelectable [static, inherited]
```

Disables selection of all instances of this type.

Reimplemented from SceneObject.

```cpp
bool PxCloth::isSelectable [static, inherited]
```

Disables selection of all instances of this type.

Reimplemented from GameBase.

```cpp
bool PhysicsDebris::isSelectable [static, inherited]
```

Disables selection of all instances of this type.

Reimplemented from GameBase.

```cpp
bool Splash::isSelectable [static, inherited]
```

Disables selection of all instances of this type.
Reimplemented from \texttt{GameBase}.

\begin{verbatim}
bool ParticleEmitter::isSelectable [static, inherited]
\end{verbatim}

Disables selection of all instances of this type.

Reimplemented from \texttt{GameBase}.

\begin{verbatim}
bool GroundCover::isSelectable [static, inherited]
\end{verbatim}

Disables selection of all instances of this type.

Reimplemented from \texttt{SceneObject}.

\begin{verbatim}
bool fxShapeReplicatedStatic::isSelectable [static, inherited]
\end{verbatim}

Disables selection of all instances of this type.

Reimplemented from \texttt{SceneObject}.

\begin{verbatim}
bool fxShapeReplicator::isSelectable [static, inherited]
\end{verbatim}

Disables selection of all instances of this type.

Reimplemented from \texttt{SceneObject}.

\begin{verbatim}
bool fxFoliageReplicator::isSelectable [static, inherited]
\end{verbatim}

Disables selection of all instances of this type.

Reimplemented from \texttt{SceneObject}.
bool Explosion::isSelectable [static, inherited]

Disables selection of all instances of this type.
Reimplemented from GameBase.

bool RenderShapeExample::isSelectable [static, inherited]

Disables selection of all instances of this type.
Reimplemented from SceneObject.

bool RenderObjectExample::isSelectable [static, inherited]

Disables selection of all instances of this type.
Reimplemented from SceneObject.

bool DecalManager::isSelectable [static, inherited]

Disables selection of all instances of this type.
Reimplemented from SceneObject.

bool SpotLight::isSelectable [static, inherited]

Disables selection of all instances of this type.
Reimplemented from LightBase.

bool StaticShape::isSelectable [static, inherited]
Disables selection of all instances of this type.

Reimplemented from "ShapeBase".

```cpp
bool PointLight::isSelectable [static, inherited]
```

Disables selection of all instances of this type.

Reimplemented from "LightBase".

```cpp
bool LightBase::isSelectable [static, inherited]
```

Disables selection of all instances of this type.

Reimplemented from "SceneObject".

Reimplemented in "PointLight", and "SpotLight".

```cpp
bool OcclusionVolume::isSelectable [static, inherited]
```

Disables selection of all instances of this type.

Reimplemented from "SceneObject".

```cpp
bool WayPoint::isSelectable [static, inherited]
```

Disables selection of all instances of this type.

Reimplemented from "MissionMarker".

```cpp
bool MissionMarker::isSelectable [static, inherited]
```
Disables selection of all instances of this type.
Reimplemented from `ShapeBase`.

Reimplemented in `WayPoint` and `SpawnSphere`.

```cpp
bool ShapeBase::isSelectable [static, inherited]
```

Disables selection of all instances of this type.
Reimplemented from `GameBase`.

Reimplemented in `MissionMarker`, `WayPoint`, `StaticShape`, `Vehicle`, `HoverVehicle`, `Player`, `AIPlayer`, `Camera`, `Item`, `SpawnSphere`, `PathCamera`, `ProximityMine`, `RigidShape`, `TurretShape`, `AITurretShape`, `FlyingVehicle`, and `WheeledVehicle`.

```cpp
bool GameBase::isSelectable [static, inherited]
```

Disables selection of all instances of this type.
Reimplemented from `SceneObject`.

Reimplemented in `ShapeBase`, `MissionMarker`, `WayPoint`, `StaticShape`, `Explosion`, `ParticleEmitter`, `Splash`, `PhysicsDebris`, `PxCloth`, `Vehicle`, `HoverVehicle`, `Player`, `AIPlayer`, `Camera`, `Debris`, `Item`, `SpawnSphere`, `PathCamera`, `Projectile`, `ProximityMine`, `RigidShape`, `Trigger`, `Lightning`, `ParticleEmitterNode`, `Precipitation`, `PhysicsShape`, `PxMultiActor`, `TurretShape`, `AITurretShape`, `FlyingVehicle`, and `WheeledVehicle`.

```cpp
bool ConvexShape::isSelectable [static, inherited]
```

Disables selection of all instances of this type.
Reimplemented from `SceneObject`.

```cpp
bool SceneObject::isSelectable [static, inherited]
```

Disables selection of all instances of this type.

Reimplemented in `ConvexShape`, `GameBase`, `ShapeBase`, `MissionMarker`, `WayPoint`, `OcclusionVolume`, `LightBase`, `PointLight`, `StaticShape`, `SpotLight`, `DecalManager`, `RenderObjectExample`, `RenderShapeExample`, `Explosion`, `fxFoliageReplicator`, `fxShapeReplicator`, `fxShapeReplicatedStatic`, `GroundCover`, `ParticleEmitter`, `Splash`, `PhysicsDebris`, `PxCloth`, `SFXSpace`, `Vehicle`, `HoverVehicle`, `BasicClouds`, `CloudLayer`, `WaterObject`, `WaterBlock`, `WaterPlane`, `Marker`, `Player`, `AIPlayer`, `Camera`, `Debris`, `GroundPlane`, `Item`, `SpawnSphere`, `PathCamera`, `PhysicalZone`, `Zone`, `Portal`, `Prefab`, `Projectile`, `ProximityMine`, `RigidShape`, `Trigger`, `TSStatic`, `RenderMeshExample`, `Lightning`, `ParticleEmitterNode`, `Precipitation`, `PhysicsForce`, `PhysicsShape`, `PxMultiActor`, `SFXEmitter`, `TurretShape`, `AITurretShape`, `FlyingVehicle`, `WheeledVehicle`, `DecalRoad`, `MeshRoad`, `River`, `ScatterSky`, `SkyBox`, `Sun`, `TimeOfDay`, `Forest`, `ForestWindEmitter`, `InteriorInstance`, and `TerrainBlock`.

```cpp
bool isWebDemo
```

**UNDOCUMENTED!**

```cpp
float $pref::PhysicsDebris::lifetimeScale
```

Scales how long `PhysicsDebris` will live before being removed.

**Note:**

A value of 0 will disable `PhysicsDebris` entirely.
int nameToID

determineToID(object)

bool PhysicsShape::noCorrections [static, inherited]

Determines if the shape will receive corrections from the server or will instead be allowed to diverge.

In the event that the client and server object positions/orientations differ and if this variable is true, the server will attempt to 'correct' the client object to keep it in sync. Otherwise, client and server objects may fall out of sync.

bool PhysicsShape::noSmoothing [static, inherited]

Determines if client-side shapes will attempt to smoothly transition to their new position after receiving a correction.

If true, shapes will immediately render at the position they are corrected to.

void physicsDebugDraw

drawDebugDraw( bool enable )

void physicsDestroy

destroy()

void physicsDestroyWorld
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>physicsDestroyWorld</code></td>
<td>Destroys the world specified by <code>worldName</code>.</td>
</tr>
<tr>
<td><code>physicsGetTimeScale</code></td>
<td>Returns the time scale.</td>
</tr>
<tr>
<td><code>physicsGetTimeScale()</code></td>
<td></td>
</tr>
<tr>
<td><code>physicsInit</code></td>
<td>Initializes the physics engine.</td>
</tr>
<tr>
<td><code>physicsInit([string library])</code></td>
<td></td>
</tr>
<tr>
<td><code>physicsInitWorld</code></td>
<td>Initializes a world with the given <code>worldName</code>.</td>
</tr>
<tr>
<td><code>physicsRestoreState</code></td>
<td>Restores the state of the physics engine.</td>
</tr>
<tr>
<td><code>physicsSetTimeScale</code></td>
<td>Sets the time scale to <code>scale</code>.</td>
</tr>
<tr>
<td><code>physicsSetTimeScale(F32 scale)</code></td>
<td></td>
</tr>
<tr>
<td><code>physicsSimulationEnabled()</code></td>
<td>Checks if the simulation is enabled.</td>
</tr>
<tr>
<td><code>physicsSimulationEnabled()</code></td>
<td></td>
</tr>
<tr>
<td><code>physicsStartSimulation</code></td>
<td>Starts the simulation for the specified <code>worldName</code>.</td>
</tr>
</tbody>
</table>
void physicsStopSimulation

physicsStopSimulation( String worldName )

void physicsStoreState

physicsStoreState()

bool physXRemoteDebuggerConnect

UNDOCUMENTED!

void physXRemoteDebuggerDisconnect

UNDOCUMENTED!

void queryAllServers

queryAllServers(...);

void queryLanServers

queryLanServers(...);

void queryMasterServer

queryMasterServer(...)
void querySingleServer

querySingleServer(address, flags);

void restartInstance

restartInstance()

void sbmDumpStats

UNDOCUMENTED!

void sbmDumpStrings

UNDOCUMENTED!

int schedule

schedule(time, refobject|0, command, <arg1...argN>)

bool setServerInfo

setServerInfo(index);

bool setShadowManager

string sShadowSystemName
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>string setShadowVizLight</code></td>
<td>UNDOCUMENTED!</td>
</tr>
<tr>
<td><code>void startHeartbeat</code></td>
<td>startHeartbeat()</td>
</tr>
<tr>
<td><code>int startPrecisionTimer</code></td>
<td><code>startPrecisionTimer(S32 id)</code> - Create and start a high resolution platform</td>
</tr>
<tr>
<td></td>
<td>timer. Returns the timer id.</td>
</tr>
<tr>
<td><code>void stopHeartbeat</code></td>
<td>stopHeartbeat();</td>
</tr>
<tr>
<td><code>int stopPrecisionTimer</code></td>
<td><code>stopPrecisionTimer(S32 id)</code> - Stop and destroy timer with the passed id.</td>
</tr>
<tr>
<td></td>
<td>Returns the elapsed milliseconds.</td>
</tr>
<tr>
<td><code>void stopServerQuery</code></td>
<td>stopServerQuery()</td>
</tr>
<tr>
<td><code>string strToPlayerName</code></td>
<td><code>strToPlayerName(string)</code></td>
</tr>
</tbody>
</table>
string TestFunction2Args

testFunction(arg1, arg2)

string testJavaScriptBridge

testBridge(arg1, arg2, arg3)

void tsUpdateImposterImages

tsUpdateImposterImages( bool forceupdate )

Copyright © GarageGames, LLC. All Rights Reserved.
Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActionMap</td>
<td>ActionMaps assign platform input events to console commands</td>
</tr>
<tr>
<td>AdvancedLightBinManager</td>
<td>Rendering Manager responsible for lighting, shadows, and global variables affecting both</td>
</tr>
<tr>
<td>AIClient</td>
<td>Simulated client driven by AI commands</td>
</tr>
<tr>
<td>AIConnection</td>
<td>Special client connection driven by an AI, rather than a human</td>
</tr>
<tr>
<td>AIPlayer</td>
<td>A Player object not controlled by conventional input, but by an AI engine</td>
</tr>
<tr>
<td>AITurretShape</td>
<td>Provides an AI controlled turret</td>
</tr>
<tr>
<td>AITurretShapeData</td>
<td>Defines properties for an AITurretShape object</td>
</tr>
<tr>
<td>ArrayObject</td>
<td>Data structure for storing indexed sequences of key/value pairs</td>
</tr>
<tr>
<td>BanList</td>
<td>Used for kicking and banning players from a server. There is only a single instance of BanList.</td>
</tr>
<tr>
<td></td>
<td>It is very important to note that you do not ever create this object in script like you would</td>
</tr>
<tr>
<td></td>
<td>other game play objects. You simply reference it via namespace</td>
</tr>
<tr>
<td>BasicClouds</td>
<td>Renders up to three layers of scrolling cloud-cover textures overhead</td>
</tr>
<tr>
<td>Camera</td>
<td>Represents a position, direction and field of view to render a scene from</td>
</tr>
<tr>
<td><strong>CameraData</strong></td>
<td>A datablock that describes a camera</td>
</tr>
<tr>
<td><strong>CloudLayer</strong></td>
<td>A layer of clouds which change shape over time and are affected by scene lighting</td>
</tr>
<tr>
<td><strong>ConsoleLogger</strong></td>
<td></td>
</tr>
<tr>
<td><strong>ConvexShape</strong></td>
<td>A renderable, collidable convex shape defined by a collection of surface planes</td>
</tr>
<tr>
<td><strong>CubemapData</strong></td>
<td>Used to create static or dynamic cubemaps</td>
</tr>
<tr>
<td><strong>CustomMaterial</strong></td>
<td>Material object which provides more control over surface properties</td>
</tr>
<tr>
<td><strong>Debris</strong></td>
<td>Base debris class. Uses the DebrisData datablock for properties of individual debris objects</td>
</tr>
<tr>
<td><strong>DebrisData</strong></td>
<td>Stores properties for an individual debris type</td>
</tr>
<tr>
<td><strong>DebugDrawer</strong></td>
<td>A debug helper for rendering debug primitives to the scene</td>
</tr>
<tr>
<td><strong>DecalData</strong></td>
<td>A datablock describing an individual decal</td>
</tr>
<tr>
<td><strong>DecalManager</strong></td>
<td>The object that manages all of the decals in the active mission</td>
</tr>
<tr>
<td><strong>DecalRoad</strong></td>
<td>A strip shaped decal defined by spine nodes which clips against Terrain objects</td>
</tr>
<tr>
<td><strong>EditorIconRegistry</strong></td>
<td>This class is used to find the correct icon file path for different SimObject class types</td>
</tr>
<tr>
<td><strong>EventManager</strong></td>
<td>Wrapper for the standard messaging system</td>
</tr>
<tr>
<td></td>
<td>The emitter for an explosion effect, with properties defined by a</td>
</tr>
<tr>
<td><strong>Explosion</strong></td>
<td><strong>ExplosionData</strong> object</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td><strong>ExplosionData</strong></td>
<td>Defines the attributes of an Explosion: particleEmitters, debris, lighting and camera shake effects</td>
</tr>
<tr>
<td><strong>FileDialog</strong></td>
<td>Base class responsible for displaying an OS file browser</td>
</tr>
<tr>
<td><strong>FileObject</strong></td>
<td>This class is responsible opening, reading, creating, and saving file contents</td>
</tr>
<tr>
<td><strong>FileStreamObject</strong></td>
<td>A wrapper around StreamObject for parsing text and data from files</td>
</tr>
<tr>
<td><strong>FlyingVehicle</strong></td>
<td>A flying vehicle</td>
</tr>
<tr>
<td><strong>FlyingVehicleData</strong></td>
<td>Defines the properties of a FlyingVehicle</td>
</tr>
<tr>
<td><strong>Forest</strong></td>
<td>Forest is a global-bounds scene object provides collision and rendering for a (.forest) data file</td>
</tr>
<tr>
<td><strong>ForestBrushElement</strong></td>
<td>Represents a type of ForestItem and parameters for how it is placed when painting with a ForestBrush that contains it</td>
</tr>
<tr>
<td><strong>ForestItemData</strong></td>
<td>Base class for defining a type of ForestItem. It does not implement loading or rendering of the shapeFile</td>
</tr>
<tr>
<td><strong>ForestWindEmitter</strong></td>
<td>Object responsible for simulating wind in a level</td>
</tr>
<tr>
<td><strong>fxFoliageReplicator</strong></td>
<td>An emitter to replicate fxFoliageItem objects across an area</td>
</tr>
<tr>
<td><strong>fxShapeReplicatedStatic</strong></td>
<td>The object definition for shapes that will be replicated across an area using an fxShapeReplicator</td>
</tr>
<tr>
<td><strong>fxShapeReplicator</strong></td>
<td>An emitter for objects to replicate across an area</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>GameBase</td>
<td>Base class for game objects which use datablocks, networking, are editable, and need to process ticks</td>
</tr>
<tr>
<td>GameBaseData</td>
<td>Scriptable, demo-able datablock. Used by GameBase objects</td>
</tr>
<tr>
<td>GameConnection</td>
<td>The game-specific subclass of NetConnection</td>
</tr>
<tr>
<td>GameTSCtrl</td>
<td>The main 3D viewport for a Torque 3D game</td>
</tr>
<tr>
<td>GFXCardProfiler</td>
<td>Provides a device independent wrapper around both the capabilities reported by the card/drivers and the exceptions recorded in various scripts</td>
</tr>
<tr>
<td>GFXCardProfilerAPI</td>
<td>This class is the interface between TorqueScript and GFXCardProfiler</td>
</tr>
<tr>
<td>GFXInit</td>
<td>Functions for tracking GFX adapters and initializing them into devices</td>
</tr>
<tr>
<td>GFXSamplerStateData</td>
<td>A sampler state used by GFXStateBlockData</td>
</tr>
<tr>
<td>GFXStateBlockData</td>
<td>A state block description for rendering</td>
</tr>
<tr>
<td>GroundCover</td>
<td>Covers the ground in a field of objects (IE: Grass, Flowers, etc)</td>
</tr>
<tr>
<td>GroundPlane</td>
<td>An infinite plane extending in all direction</td>
</tr>
<tr>
<td>GuiArrayCtrl</td>
<td>A container that scrolls its child control up over time</td>
</tr>
<tr>
<td>GuiAutoScrollCtrl</td>
<td>A control that renders a skinned border specified in its profile</td>
</tr>
<tr>
<td>GuiBitmapBorderCtrl</td>
<td>A button that renders its various states (mouse over, pushed, etc.) from separate bitmaps</td>
</tr>
<tr>
<td>Class Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>GuiBitmapButtonTextCtrl</td>
<td>An extension of GuiBitmapButtonCtrl that additionally renders a text label on the bitmapped button</td>
</tr>
<tr>
<td>GuiBitmapCtrl</td>
<td>A gui control that is used to display an image</td>
</tr>
<tr>
<td>GuiBorderButtonCtrl</td>
<td>A push button that renders only a border</td>
</tr>
<tr>
<td>GuiBubbleTextCtrl</td>
<td>A single-line text control that displays its text in a multi-line popup when clicked</td>
</tr>
<tr>
<td>GuiButtonBaseCtrl</td>
<td>The base class for the various button controls</td>
</tr>
<tr>
<td>GuiButtonCtrl</td>
<td>The most widely used button class</td>
</tr>
<tr>
<td>GuiCanvas</td>
<td>A canvas on which rendering occurs</td>
</tr>
<tr>
<td>GuiCheckBoxCtrl</td>
<td>A named checkbox that can be toggled on and off</td>
</tr>
<tr>
<td>GuiChunkedBitmapCtrl</td>
<td>This is a control that will render a specified bitmap or a bitmap specified in a referenced variable</td>
</tr>
<tr>
<td>GuiClockHud</td>
<td>Basic HUD clock. Displays the current simulation time offset from some base</td>
</tr>
<tr>
<td>GuiConsole</td>
<td>The on-screen, in-game console. Calls getLog() to get the on-screen console entries, then renders them as needed</td>
</tr>
<tr>
<td>GuiConsoleEditCtrl</td>
<td>Text entry element of a GuiConsole</td>
</tr>
<tr>
<td>GuiContainer</td>
<td>Brief Desc</td>
</tr>
<tr>
<td>GuiControl</td>
<td>Base class for all Gui control objects</td>
</tr>
<tr>
<td>GuiControlArrayControl</td>
<td>Brief Desc</td>
</tr>
</tbody>
</table>

A collection of properties that
<table>
<thead>
<tr>
<th>Class Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GuiControlProfile</td>
<td>determine control behavior and rendering</td>
</tr>
<tr>
<td>GuiCrossHairHud</td>
<td>Basic cross hair hud. Reacts to state of control object. Also displays health bar for named objects under the cross hair</td>
</tr>
<tr>
<td>GuiCursor</td>
<td>Acts as a skin for the cursor, where each GuiCursor object can have its own look and click-zone</td>
</tr>
<tr>
<td>GuiDirectoryFileListCtrl</td>
<td>A control that displays a list of files from within a single directory in the game file system</td>
</tr>
<tr>
<td>GuiDragAndDropControl</td>
<td>A container control that can be used to implement drag&amp;drop behavior</td>
</tr>
<tr>
<td>GuiDynamicCtrlArrayControl</td>
<td>A container that arranges children into a grid</td>
</tr>
<tr>
<td>GuiFadeinBitmapCtrl</td>
<td>A GUI control which renders a black square over a bitmap image. The black square will fade out, then fade back in after a determined time. This control is especially useful for transitions and splash screens</td>
</tr>
<tr>
<td>GuiFrameSetCtrl</td>
<td>A gui control allowing a window to be subdivided into panes, each of which displays a gui control child of the GuiFrameSetCtrl</td>
</tr>
<tr>
<td>GuiGameListMenuCtrl</td>
<td>A base class for cross platform menu controls that are gamepad friendly</td>
</tr>
<tr>
<td>GuiGameListMenuProfile</td>
<td>A GuiControlProfile with additional fields specific to GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>GuiGameListOptionsCtrl</td>
<td>A control for showing pages of options that are gamepad friendly</td>
</tr>
<tr>
<td><strong>GuiGameListOptionsProfile</strong></td>
<td>A <strong>GuiControlProfile</strong> with additional fields specific to <strong>GuiGameListOptionsCtrl</strong></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>GuiGraphCtrl</strong></td>
<td>A control that plots one or more curves in a chart</td>
</tr>
<tr>
<td><strong>GuiHealthBarHud</strong></td>
<td>A basic health bar. Shows the damage value of the current PlayerObjectType control object</td>
</tr>
<tr>
<td><strong>GuiHealthTextHud</strong></td>
<td>Shows the health or energy value of the current PlayerObjectType control object</td>
</tr>
<tr>
<td><strong>GuiIconButtonCtrl</strong></td>
<td>Draws the bitmap within a special button control. Only a single bitmap is used and the button will be drawn in a highlighted mode when the mouse hovers over it or when it has been clicked</td>
</tr>
<tr>
<td><strong>GuiInputCtrl</strong></td>
<td>A control that locks the mouse and reports all keyboard input events to script</td>
</tr>
<tr>
<td><strong>GuiListBoxCtrl</strong></td>
<td>A list of text items</td>
</tr>
<tr>
<td><strong>GuiMenuBar</strong></td>
<td>GUI Control which displays a horizontal bar with individual drop-down menu items. Each menu item may also have submenu items</td>
</tr>
<tr>
<td><strong>GuiMessageVectorCtrl</strong></td>
<td>A chat HUD control that displays messages from a <strong>MessageVector</strong></td>
</tr>
<tr>
<td><strong>GuiMLTextCtrl</strong></td>
<td>A text control that uses the Gui Markup Language ('ML') tags to dynamically change the text</td>
</tr>
<tr>
<td><strong>GuiMLTextEditCtrl</strong></td>
<td>A text entry control that accepts the Gui Markup Language ('ML') tags and multiple lines</td>
</tr>
<tr>
<td></td>
<td>Used to overlaps a 'hot region' where you want to catch inputs</td>
</tr>
<tr>
<td>Class</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>GuiMouseEventCtrl</td>
<td>with and have specific events occur based on individual callbacks</td>
</tr>
<tr>
<td>GuiObjectView</td>
<td>GUI control which displays a 3D model</td>
</tr>
<tr>
<td>GuiPaneControl</td>
<td>A collapsible pane control</td>
</tr>
<tr>
<td>GuiPanel</td>
<td>The GuiPanel panel is a container that when opaque will draw a left to right gradient using its profile fill and highlight colors</td>
</tr>
<tr>
<td>GuiPopUpMenuCtrl</td>
<td>A control that allows to select a value from a drop-down list</td>
</tr>
<tr>
<td>GuiPopUpMenuCtrlEx</td>
<td>A control that allows to select a value from a drop-down list</td>
</tr>
<tr>
<td>GuiProgressBitmapCtrl</td>
<td>A horizontal progress bar rendered from a repeating image</td>
</tr>
<tr>
<td>GuiProgressCtrl</td>
<td>GUI Control which displays a horizontal bar which increases as the progress value of 0.0 - 1.0 increases</td>
</tr>
<tr>
<td>GuiRadioCtrl</td>
<td>A button based around the radio concept</td>
</tr>
<tr>
<td>GuiRolloutCtrl</td>
<td>A container that shows a single child with an optional header bar that can be used to collapse and expand the rollout</td>
</tr>
<tr>
<td>GuiScriptNotifyCtrl</td>
<td>A control which adds several reactions to other GUIs via callbacks</td>
</tr>
<tr>
<td>GuiScrollCtrl</td>
<td>A container that allows to view one or more possibly larger controls inside its area by providing horizontal and/or vertical scroll bars</td>
</tr>
<tr>
<td>GuiSeparatorCtrl</td>
<td>A control that renders a horizontal or vertical separator with an</td>
</tr>
<tr>
<td>Class Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>GuiShapeNameHud</td>
<td>Displays name and damage of <code>ShapeBase</code> objects in its bounds. Must be a child of a <code>GuiTSCtrl</code> and a server connection must be present.</td>
</tr>
<tr>
<td>GuiSliderCtrl</td>
<td>A control that displays a value between its minimal and maximal bounds using a slider placed on a vertical or horizontal axis.</td>
</tr>
<tr>
<td>GuiSpeedometerHud</td>
<td>Displays the speed of the current <code>Vehicle</code> based control object.</td>
</tr>
<tr>
<td>GuiSplitContainer</td>
<td>A container that splits its area between two child controls.</td>
</tr>
<tr>
<td>GuiStackControl</td>
<td>A container that stacks its children horizontally or vertically.</td>
</tr>
<tr>
<td>GuiSwatchButtonCtrl</td>
<td>A button that is used to represent color; often used in correlation with a color picker.</td>
</tr>
<tr>
<td>GuiTabBookCtrl</td>
<td>A container</td>
</tr>
<tr>
<td>GuiTabPageCtrl</td>
<td>A single page in a <code>GuiTabBookCtrl</code></td>
</tr>
<tr>
<td>GuiTextCtrl</td>
<td>GUI control object this displays a single line of text, without TorqueML.</td>
</tr>
<tr>
<td>GuiTextEditCtrl</td>
<td>A component that places a text entry box on the screen.</td>
</tr>
<tr>
<td>GuiTextEditSliderBitmapCtrl</td>
<td>GUI Control which displays a numerical value which can be increased or decreased using a pair of bitmap up/down buttons.</td>
</tr>
<tr>
<td>GuiTextEditSliderCtrl</td>
<td>GUI Control which displays a numerical value which can be increased or decreased using a pair of arrows.</td>
</tr>
<tr>
<td>GuiTabBookCtrl</td>
<td>A container</td>
</tr>
<tr>
<td>GuiTabPageCtrl</td>
<td>A single page in a <code>GuiTabBookCtrl</code></td>
</tr>
<tr>
<td>Class/Symbol</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>GuiTextListCtrl</td>
<td>Text. Text items in the list can be individually selected</td>
</tr>
<tr>
<td>GuiTheoraCtrl</td>
<td>A control to playing Theora videos</td>
</tr>
<tr>
<td>GuiTickCtrl</td>
<td>Brief Description</td>
</tr>
<tr>
<td>GuiToggleButtonCtrl</td>
<td>Deprecated gui control</td>
</tr>
<tr>
<td>GuiTreeViewCtrl</td>
<td>Hierarchical list of text items with optional icons</td>
</tr>
<tr>
<td>GuiTSCtrl</td>
<td>Abstract base class for controls that render 3D scenes</td>
</tr>
<tr>
<td>GuiWindowCtrl</td>
<td>A window with a title bar and an optional set of buttons</td>
</tr>
<tr>
<td>HoverVehicle</td>
<td>A hovering vehicle</td>
</tr>
<tr>
<td>HoverVehicleData</td>
<td>Defines the properties of a HoverVehicle</td>
</tr>
<tr>
<td>HTTPObject</td>
<td>Allows communications between the game and a server using HTTP</td>
</tr>
<tr>
<td>InteriorInstance</td>
<td>Object used to represent buildings and other architectural structures (legacy)</td>
</tr>
<tr>
<td>Item</td>
<td>Base Item class. Uses the ItemData datablock for common properties</td>
</tr>
<tr>
<td>ItemData</td>
<td>Stores properties for an individual Item type</td>
</tr>
<tr>
<td>LangTable</td>
<td>Provides the code necessary to handle the low level management of the string tables for localization</td>
</tr>
<tr>
<td>LevelInfo</td>
<td>Stores and controls the rendering and status information for a game level</td>
</tr>
<tr>
<td>LightAnimData</td>
<td>A datablock which defines and performs light animation, such as rotation, brightness fade, and colorization</td>
</tr>
<tr>
<td>Class</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>LightBase</td>
<td>This is the base class for light objects</td>
</tr>
<tr>
<td>LightDescription</td>
<td>A helper datablock used by classes (such as shapebase) that submit lights to the scene but do not use actual &quot;LightBase&quot; objects</td>
</tr>
<tr>
<td>LightFlareData</td>
<td>Defines a light flare effect usable by scene lights</td>
</tr>
<tr>
<td>Lightning</td>
<td>An emitter for lightning bolts</td>
</tr>
<tr>
<td>LightningData</td>
<td>Common data for a Lightning emitter object</td>
</tr>
<tr>
<td>LightningStrikeEvent</td>
<td>Network event that triggers a lightning strike on the client when it is received</td>
</tr>
<tr>
<td>Marker</td>
<td>A single joint, or knot, along a path. Should be stored inside a Path container object. A path markers can be one of three primary movement types: &quot;normal&quot;, &quot;Position Only&quot;, or &quot;Kink&quot;</td>
</tr>
<tr>
<td>Material</td>
<td>A material in Torque 3D is a data structure that describes a surface</td>
</tr>
<tr>
<td>MeshRoad</td>
<td>A strip of rectangular mesh segments defined by a 3D spline for prototyping road-shaped objects in your scene</td>
</tr>
<tr>
<td>Message</td>
<td>Base class for messages</td>
</tr>
<tr>
<td>MessageForwarder</td>
<td>Forward messages from one queue to another</td>
</tr>
<tr>
<td>MessageVector</td>
<td>Store a list of chat messages</td>
</tr>
<tr>
<td>MissionArea</td>
<td>Level object which defines the boundaries of the level</td>
</tr>
<tr>
<td>MissionMarker</td>
<td>This is a base class for all &quot;marker&quot; related objects. It is a 3D representation of a point in the level</td>
</tr>
<tr>
<td><strong>MissionMarkerData</strong></td>
<td>A very basic class containing information used by MissionMarker objects for rendering</td>
</tr>
<tr>
<td>-----------------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>NetConnection</strong></td>
<td>Provides the basis for implementing a multiplayer game protocol</td>
</tr>
<tr>
<td><strong>NetObject</strong></td>
<td>Superclass for all ghostable networked objects</td>
</tr>
<tr>
<td><strong>OcclusionVolume</strong></td>
<td>An invisible shape that causes objects hidden from view behind it to not be rendered</td>
</tr>
<tr>
<td><strong>OpenFileDialog</strong></td>
<td>Derived from FileDialog, this class is responsible for opening a file browser with the intention of opening a file</td>
</tr>
<tr>
<td><strong>OpenFolderDialog</strong></td>
<td>OS level dialog used for browsing folder structures</td>
</tr>
<tr>
<td><strong>ParticleData</strong></td>
<td>Contains information for how specific particles should look and react including particle colors, particle imagemap, acceleration value for individual particles and spin information</td>
</tr>
<tr>
<td><strong>ParticleEmitter</strong></td>
<td>This object is responsible for spawning particles</td>
</tr>
<tr>
<td><strong>ParticleEmitterData</strong></td>
<td>Defines particle emission properties such as ejection angle, period and velocity for a ParticleEmitter</td>
</tr>
<tr>
<td><strong>ParticleEmitterNode</strong></td>
<td>A particle emitter object that can be positioned in the world and dynamically enabled or disabled</td>
</tr>
<tr>
<td><strong>ParticleEmitterNodeData</strong></td>
<td>Contains additional data to be associated with a ParticleEmitterNode</td>
</tr>
</tbody>
</table>

A spline along which various
<p>| <strong>Path</strong> | objects can move along. The spline object acts like a container for Marker objects, which make up the joints, or knots, along the path. Paths can be assigned a speed, can be looping or non-looping. Each of a path's markers can be one of three primary movement types: &quot;normal&quot;, &quot;Position Only&quot;, or &quot;Kink&quot; |
| <strong>PathCamera</strong> | A camera that moves along a path. The camera can then be made to travel along this path forwards or backwards |
| <strong>PathCameraData</strong> | General interface to control a PathCamera object from the script level |
| <strong>PfxVis</strong> | Singleton class that exposes ConsoleStaticFunctions for debug visualizing PostEffects |
| <strong>PhysicalZone</strong> | Physical Zones are areas that modify the player's gravity and/or velocity and/or applied force |
| <strong>PhysicsDebris</strong> | Represents one or more rigid bodies defined in a single mesh file with a limited lifetime |
| <strong>PhysicsDebrisData</strong> | Defines the properties of a PhysicsDebris object |
| <strong>PhysicsForce</strong> | Helper object for gameplay physical forces |
| <strong>PhysicsShape</strong> | Represents a destructible physical object simulated through the plugin system |
| <strong>PhysicsShapeData</strong> | Defines the properties of a PhysicsShape |
| <strong>Player</strong> | A client-controlled player character |
| <strong>PlayerData</strong> | Defines properties for a <strong>Player</strong> object |
| <strong>PointLight</strong> | Lighting object that radiates light in all directions |
| <strong>Portal</strong> | An object that provides a &quot;window&quot; into a zone, allowing a viewer to see what's rendered in the zone |
| <strong>PostEffect</strong> | A fullscreen shader effect |
| <strong>Precipitation</strong> | Defines a precipitation based storm (rain, snow, etc) |
| <strong>PrecipitationData</strong> | Defines the droplets used in a storm (raindrops, snowflakes, etc) |
| <strong>Prefab</strong> | A collection of arbitrary objects which can be allocated and manipulated as a group |
| <strong>Projectile</strong> | Base projectile class. Uses the <strong>ProjectileData</strong> class for properties of individual projectiles |
| <strong>ProjectileData</strong> | Stores properties for an individual projectile type |
| <strong>ProximityMine</strong> | A simple proximity mine |
| <strong>ProximityMineData</strong> | Stores common properties for a <strong>ProximityMine</strong> |
| <strong>PxCloth</strong> | Rectangular patch of cloth simulated by PhysX |
| <strong>PxMaterial</strong> | Defines a PhysX material assignable to a <strong>PxMaterial</strong> |
| <strong>PxMultiActor</strong> | Represents a destructible physical object simulated using PhysX |
| <strong>PxMultiActorData</strong> | Defines the properties of a type of <strong>PxMultiActor</strong> |
| <strong>RadialImpulseEvent</strong> | Creates a physics-based impulse effect from a defined central point and magnitude |
| | A datablock which defines |</p>
<table>
<thead>
<tr>
<th><strong>ReflectorDesc</strong></th>
<th>performance and quality properties for dynamic reflections</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RenderBinManager</strong></td>
<td>The abstract base for all render bins</td>
</tr>
<tr>
<td><strong>RenderFormatToken</strong></td>
<td>Used to change the render target format when rendering in AL</td>
</tr>
<tr>
<td><strong>RenderGlowMgr</strong></td>
<td>A render bin for the glow pass</td>
</tr>
<tr>
<td><strong>RenderImposterMgr</strong></td>
<td>A render bin for batch rendering imposters</td>
</tr>
<tr>
<td><strong>RenderMeshExample</strong></td>
<td>An example scene object which renders a mesh</td>
</tr>
<tr>
<td><strong>RenderMeshMgr</strong></td>
<td>A render bin for mesh rendering</td>
</tr>
<tr>
<td><strong>RenderObjectExample</strong></td>
<td>An example scene object which renders using a callback</td>
</tr>
<tr>
<td><strong>RenderObjectMgr</strong></td>
<td>A render bin which uses object callbacks for rendering</td>
</tr>
<tr>
<td><strong>RenderOcclusionMgr</strong></td>
<td>A render bin which renders occlusion query requests</td>
</tr>
<tr>
<td><strong>RenderParticleMgr</strong></td>
<td>A render bin which renders particle geometry</td>
</tr>
<tr>
<td><strong>RenderPassManager</strong></td>
<td>A grouping of render bin managers which forms a render pass</td>
</tr>
<tr>
<td><strong>RenderPassStateBin</strong></td>
<td>A non-rendering render bin used to enable/disable a</td>
</tr>
<tr>
<td></td>
<td><strong>RenderPassStateToken</strong></td>
</tr>
<tr>
<td><strong>RenderPassStateToken</strong></td>
<td>Abstract base class for <strong>RenderFormatToken</strong>, used to</td>
</tr>
<tr>
<td></td>
<td>manipulate what goes on in the render manager</td>
</tr>
<tr>
<td><strong>RenderPrePassMgr</strong></td>
<td>The render bin which performs a z+normals prepass used in</td>
</tr>
<tr>
<td></td>
<td>Advanced Lighting</td>
</tr>
<tr>
<td><strong>RenderShapeExample</strong></td>
<td>An example scene object which renders a DTS</td>
</tr>
<tr>
<td>Class Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RenderTerrainMgr</td>
<td>A render bin for terrain mesh rendering</td>
</tr>
<tr>
<td>RenderTexTargetBinManager</td>
<td>An abstract base class for render bin managers that render to a named texture target</td>
</tr>
<tr>
<td>RenderTranslucentMgr</td>
<td>A render bin for rendering translucent meshes</td>
</tr>
<tr>
<td>RigidShape</td>
<td>Implements rigid-body physics for DTS objects in the world</td>
</tr>
<tr>
<td>RigidShapeData</td>
<td>Defines the physics properties for an individual RigidShapeData physics object</td>
</tr>
<tr>
<td>River</td>
<td>A water volume defined by a 3D spline</td>
</tr>
<tr>
<td>SaveFileDialog</td>
<td>Derived from FileDialog, this class is responsible for opening a file browser with the intention of saving a file</td>
</tr>
<tr>
<td>ScatterSky</td>
<td>Represents both the sun and sky for scenes with a dynamic time of day</td>
</tr>
<tr>
<td>SceneObject</td>
<td>A networkable object that exists in the 3D world</td>
</tr>
<tr>
<td>ScriptGroup</td>
<td>Essentially a SimGroup, but with onAdd and onRemove script callbacks</td>
</tr>
<tr>
<td>ScriptMsgListener</td>
<td>Script accessible version of Dispatcher::IMessageListener. Often used in conjunction with EventManager</td>
</tr>
<tr>
<td>ScriptObject</td>
<td>A script-level OOP object which allows binding of a class, superClass and arguments along with declaration of methods</td>
</tr>
<tr>
<td>SFXAmbience</td>
<td>A datablock that describes an ambient sound space</td>
</tr>
<tr>
<td>Term</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SFXController</td>
<td>A sound source that drives multi-source playback</td>
</tr>
<tr>
<td>SFXDescription</td>
<td>A description for how a sound should be played</td>
</tr>
<tr>
<td>SFXEmitter</td>
<td>An invisible 3D object that emits sound</td>
</tr>
<tr>
<td>SFXEnvironment</td>
<td>Description of a reverb environment</td>
</tr>
<tr>
<td>SFXFMODEvent</td>
<td>A playable sound event in an FMOD Designer audio project</td>
</tr>
<tr>
<td>SFXFMODEventGroup</td>
<td>A group of events in an imported FMOD Designer project</td>
</tr>
<tr>
<td>SFXFMODEventSource</td>
<td>A sound source controller playing an FMOD Designer event (SFXFMODEvent)</td>
</tr>
<tr>
<td>SFXFMODProject</td>
<td>An FMOD Designer project loaded into Torque</td>
</tr>
<tr>
<td>SFXParameter</td>
<td>A sound channel value that can be bound to multiple sound sources</td>
</tr>
<tr>
<td>SFXPlayList</td>
<td>A datablock describing a playback pattern of sounds</td>
</tr>
<tr>
<td>SFXProfile</td>
<td>Encapsulates a single sound file for playback by the sound system</td>
</tr>
<tr>
<td>SFXSound</td>
<td>A sound controller that directly plays a single sound file</td>
</tr>
<tr>
<td>SFXSource</td>
<td>Playback controller for a sound source</td>
</tr>
<tr>
<td>SFXSpace</td>
<td>A volume in space that defines an ambient sound zone</td>
</tr>
<tr>
<td>SFXState</td>
<td>A boolean switch used to modify playlist behavior</td>
</tr>
<tr>
<td>SFXTrack</td>
<td>Abstract base class for sound data that can be played back by the sound system</td>
</tr>
<tr>
<td></td>
<td>Special type of data block that</td>
</tr>
<tr>
<td><strong>ShaderData</strong></td>
<td>stores information about a handwritten shader</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td><strong>ShapeBase</strong></td>
<td>A scriptable, renderable shape</td>
</tr>
<tr>
<td><strong>ShapeBaseData</strong></td>
<td>Defines properties for a ShapeBase object</td>
</tr>
<tr>
<td><strong>ShapeBaseImageData</strong></td>
<td>Represents geometry to be mounted to a ShapeBase object</td>
</tr>
<tr>
<td><strong>SimDataBlock</strong></td>
<td></td>
</tr>
<tr>
<td><strong>SimGroup</strong></td>
<td>A collection of SimObjects that are owned by the group</td>
</tr>
<tr>
<td><strong>SimObject</strong></td>
<td>Base class for almost all objects involved in the simulation</td>
</tr>
<tr>
<td><strong>SimpleMessageEvent</strong></td>
<td>A very simple example of a network event</td>
</tr>
<tr>
<td><strong>SimpleNetObject</strong></td>
<td>A very simple example of a class derived from NetObject</td>
</tr>
<tr>
<td><strong>SimSet</strong></td>
<td>A collection of SimObjects</td>
</tr>
<tr>
<td><strong>SimXMLDocument</strong></td>
<td>File I/O object used for creating, reading, and writing XML documents</td>
</tr>
<tr>
<td><strong>SkyBox</strong></td>
<td>Represents the sky with an artist-created cubemap</td>
</tr>
<tr>
<td><strong>SpawnSphere</strong></td>
<td>This class is used for creating any type of game object, assigning it a class, datablock, and other properties when it is spawned</td>
</tr>
<tr>
<td><strong>Splash</strong></td>
<td>Manages the ring used for a Splash effect</td>
</tr>
<tr>
<td><strong>SplashData</strong></td>
<td>Acts as the physical point in space in white a Splash is created from</td>
</tr>
<tr>
<td><strong>SpotLight</strong></td>
<td>Lighting object which emits conical light in a direction</td>
</tr>
<tr>
<td><strong>StaticShape</strong></td>
<td>The most basic 3D shape with a datablock available in Torque 3D</td>
</tr>
<tr>
<td>Class/Interface</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>StaticShapeData</strong></td>
<td>The most basic <em>ShapeBaseData</em> derived shape datablock available in Torque 3D</td>
</tr>
<tr>
<td><strong>StreamObject</strong></td>
<td>Base class for working with streams</td>
</tr>
<tr>
<td><strong>Sun</strong></td>
<td>A global light affecting your entire scene and optionally renders a corona effect</td>
</tr>
<tr>
<td><strong>TCPOObject</strong></td>
<td>Allows communications between the game and a server using TCP/IP protocols</td>
</tr>
<tr>
<td><strong>TerrainBlock</strong></td>
<td>Represent a terrain object in a Torque 3D level</td>
</tr>
<tr>
<td><strong>TerrainMaterial</strong></td>
<td>Originizes the material settings for a single terrain material layer</td>
</tr>
<tr>
<td><strong>TimeOfDay</strong></td>
<td>Environmental object that triggers a day/night cycle in level</td>
</tr>
<tr>
<td><strong>Trigger</strong></td>
<td>A <em>Trigger</em> is a volume of space that initiates script callbacks when objects pass through the <em>Trigger</em></td>
</tr>
<tr>
<td><strong>TriggerData</strong></td>
<td>Defines shared properties for <em>Trigger</em> objects</td>
</tr>
<tr>
<td><strong>TSForestItemData</strong></td>
<td>Concrete implementation of <em>ForestItemData</em> which loads and renders dts format shapeFiles</td>
</tr>
<tr>
<td><strong>TSShapeConstructor</strong></td>
<td>An object used to modify a DTS or COLLADA shape model after it has been loaded by Torque</td>
</tr>
<tr>
<td><strong>TSStatic</strong></td>
<td>A static object derived from a 3D model file and placed within the game world</td>
</tr>
<tr>
<td><strong>TurretShape</strong></td>
<td>Base turret class</td>
</tr>
<tr>
<td><strong>TurretShapeData</strong></td>
<td>Defines properties for a <em>TurretShape</em> object</td>
</tr>
<tr>
<td></td>
<td>Base functionality shared by all</td>
</tr>
<tr>
<td><strong>Vehicle</strong></td>
<td>Vehicles (FlyingVehicle, HoverVehicle, WheeledVehicle)</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------------------------------------------</td>
</tr>
<tr>
<td><strong>VehicleData</strong></td>
<td>Base properties shared by all Vehicles (FlyingVehicle, HoverVehicle, WheeledVehicle)</td>
</tr>
<tr>
<td><strong>WaterBlock</strong></td>
<td>A block shaped water volume defined by a 3D scale and orientation</td>
</tr>
<tr>
<td><strong>WaterObject</strong></td>
<td>Abstract base class for representing a body of water</td>
</tr>
<tr>
<td><strong>WaterPlane</strong></td>
<td>Represents a large body of water stretching to the horizon in all directions</td>
</tr>
<tr>
<td><strong>WayPoint</strong></td>
<td>Special type of marker, distinguished by a name and team ID number</td>
</tr>
<tr>
<td><strong>WheeledVehicle</strong></td>
<td>A wheeled vehicle</td>
</tr>
<tr>
<td><strong>WheeledVehicleData</strong></td>
<td>Defines the properties of a WheeledVehicle</td>
</tr>
<tr>
<td><strong>WheeledVehicleSpring</strong></td>
<td>Defines the properties of a WheeledVehicle spring</td>
</tr>
<tr>
<td><strong>WheeledVehicleTire</strong></td>
<td>Defines the properties of a WheeledVehicle tire</td>
</tr>
<tr>
<td><strong>ZipObject</strong></td>
<td>Provides access to a zip file</td>
</tr>
<tr>
<td><strong>Zone</strong></td>
<td>An object that represents an interior space</td>
</tr>
</tbody>
</table>
ActionMap Class Reference
[Input Management]

ActionMaps assign platform input events to console commands.

More...

Inheritance diagram for ActionMap:

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bool</code></td>
<td><code>bind</code> (string device, string action, string command)</td>
<td>Associates a function to an input event.</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><code>bind</code> (string device, string action, string flag, string deadZone, string scale, string command)</td>
<td>Associates a function and input parameters to an input event.</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><code>bindCmd</code> (string device, string action, string makeCmd, string breakCmd=&quot;&quot;)</td>
<td>Associates a make command and optional break command to a specified input device action.</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><code>bindObj</code> (string device, string action, string command, SimObjectID object)</td>
<td>Associates a function to an input event for a specified class or object.</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><code>bindObj</code> (string device, string action, string flag, string deadZone, string scale, string command, SimObjectID object)</td>
<td>Associates a function to an input event for a specified class or object.</td>
</tr>
<tr>
<td><code>string</code></td>
<td><code>getBinding</code> (string command)</td>
<td>Gets the <code>ActionMap</code> binding for the specified command.</td>
</tr>
<tr>
<td><code>string</code></td>
<td><code>getCommand</code> (string device, string action)</td>
<td>Gets <code>ActionMap</code> command for the device and action.</td>
</tr>
<tr>
<td><code>string</code></td>
<td><code>getDeadZone</code> (string device, string action)</td>
<td>Gets the Dead zone for the specified device and action.</td>
</tr>
<tr>
<td><code>float</code></td>
<td><code>getScale</code> (string device, string action)</td>
<td>Get any scaling on the specified device and action.</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><code>isInverted</code> (string device, string action)</td>
<td>Determines if the specified device and action is inverted.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>pop</code> ()</td>
<td></td>
</tr>
</tbody>
</table>
Pop the **ActionMap** off the ActionMap stack.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void push ()</code></td>
<td>Push the <strong>ActionMap</strong> onto the ActionMap stack.</td>
</tr>
<tr>
<td><code>void save (string fileName=NULL, bool append=false)</code></td>
<td>Saves the <strong>ActionMap</strong> to a file or dumps it to the console.</td>
</tr>
<tr>
<td><code>bool unbind (string device, string action)</code></td>
<td>Removes the binding on an input device and action.</td>
</tr>
<tr>
<td><code>bool unbindObj (string device, string action, string obj)</code></td>
<td>Remove any object-binding on an input device and action.</td>
</tr>
</tbody>
</table>
**Detailed Description**

ActionMaps assign platform input events to console commands.

Any platform input event can be bound in a single, generic way. In theory, the game doesn't need to know if the event came from the keyboard, mouse, joystick or some other input device. This allows users of the game to map keys and actions according to their own preferences. Game action maps are arranged in a stack for processing so individual parts of the game can define specific actions. For example, when the player jumps into a vehicle it could push a vehicle action map and pop the default player action map.
Creating an ActionMap

The input system allows for the creation of multiple ActionMaps, so long as they have unique names and do not already exist. It's a simple three step process.

1. Check to see if the ActionMap exists 2. Delete it if it exists 3. Instantiate the ActionMap

The following is an example of how to create a new ActionMap:

Example:

```java
if ( isObject( moveMap ) )
    moveMap.delete();
new ActionMap(moveMap);
```
Binding Functions

Once you have created an ActionMap, you can start binding functionality to events. Currently, Torque 3D supports the following devices out of the box:

Mouse

Keyboard

Joystick/Gamepad

Xbox 360 Controller

The two most commonly used binding methods are bind() and bindCmd(). Both are similar in that they will bind functionality to a device and event, but different in how the event is interpreted. With bind(), you specify a device, action to bind, then a function to be called when the event happens.

Example:

```plaintext
// Simple function that prints to console
// %val - Sent by the device letting the user
// if an input was pressed (true) or released
function testInput(%val)
{
    if(%val)
        echo("Key is down");
    else
        echo("Key was released");
}

// Bind the 'K' key to the testInput function
moveMap.bind(keyboard, "k", testInput);
```

bindCmd is an alternative method for binding commands. This
function is similar to `bind()`, except two functions are set to be called when the event is processed.

One will be called when the event is activated (input down), while the other is activated when the event is broken (input release). When using `bindCmd()`, pass the functions as strings rather than the function names.

Example:

```javascript
// Print to the console when the spacebar is pressed
function onSpaceDown()
{
    echo("Space bar down!");
}

// Print to the console when the spacebar is released
function onSpaceUp()
{
    echo("Space bar up!");
}

// Bind the commands onSpaceDown and onSpaceUp to spacebar events
moveMap.bindCmd(keyboard, "space", "onSpaceDown();";
```
Switching ActionMaps

Let's say you want to have different ActionMaps activated based on game play situations. A classic example would be first person shooter controls and racing controls in the same game. On foot, spacebar may cause your player to jump. In a vehicle, it may cause some kind of "turbo charge". You simply need to push/pop the ActionMaps appropriately:

First, create two separate ActionMaps:

Example:

```
// Create the two ActionMaps
if ( isObject( moveMap ) )
    moveMap.delete();
new ActionMap(moveMap);

if ( isObject( carMap ) )
carMap.delete();
new ActionMap(carMap);
```

Next, create the two separate functions. Both will be bound to spacebar, but not the same ActionMap:

Example:

```
// Print to the console the player is jump
function playerJump(%val)
{
    if(%val)
        echo("Player jumping!");
}

// Print to the console the vehicle is charging
function turboCharge()
```
You are now ready to bind functions to your ActionMaps' devices:

**Example:**

```javascript
// Bind the spacebar to the playerJump function
// when moveMap is the active ActionMap
moveMap.bind(keyboard, "space", playerJump);

// Bind the spacebar to the turboCharge function
// when carMap is the active ActionMap
carMap.bind(keyboard, "space", turboCharge);
```

Finally, you can use the `push()` and `pop()` commands on each `ActionMap` to toggle activation. To activate an `ActionMap`, use `push()`:

**Example:**

```javascript
// Make moveMap the active action map
// You should now be able to activate playerJump
moveMap.push();
```

To switch ActionMaps, first `pop()` the old one. Then you can `push()` the new one:

**Example:**

```javascript
// Deactivate moveMap
moveMap.pop();

// Activate carMap
carMap.push();
```
bool ActionMap::bind(string device,
                    string action,
                    string command)

Associates a function to an input event.

When the input event is raised, the specified function will be called.

Parameters:
- **device**: The input device, such as mouse or keyboard.
- **action**: The input event, such as space, button0, etc.
- **command**: The function to bind to the action. Function must have a single boolean argument.

Returns:
True if the binding was successful, false if the device was unknown or description failed.

Example:

```cpp
// Simple function that prints to console
// %val - Sent by the device letting the user know if an input was pressed (true) or released
function testInput(%val)
{
    if(%val)
        echo("Key is down");
    else
        echo("Key was released");
}
```
Associates a function and input parameters to an input event.

When the input event is raised, the specified function will be called. Modifier flags may be specified to process dead zones, input inversion, and more.

Valid modifier flags:

- **R** - Input is Ranged.
- **S** - Input is Scaled.
- **I** - Input is inverted.
- **D** - Dead zone is present.
- **N** - Input should be re-fit to a non-linear scale.

**Parameters:**

- **device**
  The input device, such as mouse or keyboard.

- **action**
  The input event, such as space, button0, etc.

- **flag**
  Modifier flag assigned during binding, letting event know there are additional parameters to consider.

- **deadZone**
  Restricted region in which device motion will not be acknowledged.

- **scale**
  Modifies the deadZone region.

- **command**
  The function bound to the action. Must take in a single argument.

```cpp
bool ActionMap::bind(string device,
                      string action,
                      string flag,
                      string deadZone,
                      string scale,
                      string command)
```
Returns:
True if the binding was successful, false if the device was unknown or description failed.

Example:

```cpp
// Simple function that adjusts the pitch
function testPitch(%val)
{
    %pitchAdj = getMouseAdjustAmount(%val);
    $mvPitch += %pitchAdj;
}

// Bind the mouse's X axis to the testPitch
// DI is flagged, meaning input is inverted
%this.bind( mouse, "xaxis", "DI", "-0.23 0.23"

bool ActionMap::bindCmd(string device, string action, string makeCmd, string breakCmd = "")

Associates a make command and optional break command to a specified input device action.

Must include parenthesis and semicolon in the make and break command strings.

Parameters:

device The device to bind to. Can be a keyboard, mouse, joystick or gamepad.
action The device action to bind to. The action is dependant upon the device. Specify a key for keyboards.
makeCmd  The command to execute when the device/action is made.
breakCmd  [optional] The command to execute when the device or action is unmade.

Returns:
True the bind was successful, false if the device was unknown or description failed.

Example:

```cpp
// Print to the console when the spacebar is pressed
function onSpaceDown()
{
   echo("Space bar down!");
}

// Print to the console when the spacebar is released
function onSpaceUp()
{
   echo("Space bar up!");
}

// Bind the commands onSpaceDown() and onSpaceUp()
moveMap.bindCmd(keyboard, "space", "onSpaceDown()");
```

```cpp
bool ActionMap::bindObj(string device,
                        string action,
                        string command,
                        SimObjectID object)
```

Associates a function to an input event for a specified class or
object.

You must specify a device, the action to bind, a function, and an object to be called when the event happens. The function specified must be set to receive a single boolean value passed.

**Parameters:**

- **device**  
  The input device, such as mouse or keyboard.
- **action**  
  The input event, such as space, button0, etc.
- **command**  
  The function bound to the action.
- **object**  
  The object or class bound to the action.

**Returns:**

True if the binding was successful, false if the device was unknown or description failed.

**Example:**

```cpp
moveMap.bindObj(keyboard, "numpad1", "rangeChange")
```

```cpp
bool ActionMap::bindObj(string device, string action, string flag, string deadZone, string scale, string command, SimObjectID object)
```

Associates a function to an input event for a specified class or object.

You must specify a device, the action to bind, a function, and an object to be called when the event happens. The function specified must be set to receive a single boolean value passed. Modifier flags may be specified to process dead zones, input inversion, and
Valid modifier flags:

- R - Input is Ranged.
- S - Input is Scaled.
- I - Input is inverted.
- D - Dead zone is present.
- N - Input should be re-fit to a non-linear scale.

Parameters:

- **device**
  The input device, such as mouse or keyboard.

- **action**
  The input event, such as space, button0, etc.

- **flag**
  Modifier flag assigned during binding, letting event know there are additional parameters to consider.

- **deadZone**
  [Required only when flag is set] Restricted region in which device motion will not be acknowledged.

- **scale**
  [Required only when flag is set] Modifies the deadZone region.

- **command**
  The function bound to the action.

- **object**
  The object or class bound to the action.

Returns:

True if the binding was successful, false if the device was unknown or description failed.

Example:

```cpp
// Bind the mouse's movement along the x-axis
// DSI is flagged, meaning input is inverted
%this.bindObj( mouse, "xaxis", "DSI", %dead 

string ActionMap::getBinding(string command )

Gets the ActionMap binding for the specified command.

Use getField() on the return value to get the device and action of
the binding.

**Parameters:**

*command* The function to search bindings for.

**Returns:**

The binding against the specified command. Returns an empty string ("") if a binding wasn’t found.

**Example:**

```cpp
// Find what the function "jump()" is bound to in moveMap
%bind = moveMap.getBinding("jump");

if (%bind !$= "")
{
    // Find out what device is used in the binding
    %device = getField(%bind, 0);

    // Find out what action (such as a key) is used in the binding
    %action = getField(%bind, 1);
}
```

**See also:**

getField

---

```cpp
string ActionMap::getCommand(string device,
                              string action)
```

Gets *ActionMap* command for the device and action.

**Parameters:**

*device* The device that was bound. Can be a keyboard, mouse, joystick or a gamepad.
The device action that was bound. The action is dependant upon the device. Specify a key for keyboards.

Returns:
The command against the specified device and action.

Example:

```plaintext
// Find what function is bound to a device
// In this example, "jump()" was assigned
%command = moveMap.getCommand("keyboard",

// Should print "jump" in the console
echo(%command)
```

```
string ActionMap::getDeadZone(string device, string action

) Gets the Dead zone for the specified device and action.

Parameters:

device The device that was bound. Can be a keyboard, mouse, joystick or a gamepad.

action The device action that was bound. The action is dependant upon the device. Specify a key for keyboards.

Returns:
The dead zone for the specified device and action. Returns "0 0" if there is no dead zone or an empty string("") if the mapping was not found.

Example:
float ActionMap::getScale(string device,
   string action)

Get any scaling on the specified device and action.

Parameters:
- **device**: The device that was bound. Can be keyboard, mouse, joystick or gamepad.
- **action**: The device action that was bound. The action is dependant upon the device. Specify a key for keyboards.

Returns:
Any scaling applied to the specified device and action.

Example:

```cpp
%scale = %moveMap.getScale( "gamepad", "thumbrx"
```

bool ActionMap::isInverted(string device,
   string action)

Determines if the specified device and action is inverted.

Should only be used for scrolling devices or gamepad/joystick axes.

Parameters:
- **device**: The device that was bound. Can be a keyboard, mouse, joystick or a gamepad.
**action** The device action that was bound. The action is dependant upon the device. Specify a key for keyboards.

**Returns:**

True if the specified device and action is inverted.

**Example:**

```%if ( moveMap.isInverted( "mouse", "xaxis" )
    echo("Mouse's xAxis is inverted");
```

```void ActionMap::pop()
```

Pop the **ActionMap** off the ActionMap stack.

Deactivates an ActionMap and removes it from the stack.

**Example:**

```// Deactivate moveMap
moveMap.pop();
```

**See also:**

**ActionMap**

```void ActionMap::push()
```

Push the **ActionMap** onto the ActionMap stack.

Activates an **ActionMap** and places it at the top of the **ActionMap** stack.

**Example:**

```// Make moveMap the active action map```
moveMap.push();

See also:
ActionMap

```cpp
void ActionMap::save(string fileName = NULL,
                      bool append = false)
```

Saves the ActionMap to a file or dumps it to the console.

**Parameters:**

- `fileName` The file path to save the ActionMap to. If a filename is not specified the ActionMap will be dumped to the console.
- `append` Whether to write the ActionMap at the end of the file or overwrite it.

**Example:**

```cpp
// Write out the actionmap into the config.cs file
moveMap.save( "scripts/client/config.cs" );
```

```cpp
bool ActionMap::unbind(string device,
                        string action)
```

Removes the binding on an input device and action.

**Parameters:**

- `device` The device to unbind from. Can be a keyboard, mouse, joystick or a gamepad.
- `action` The device action to unbind from. The action is dependant upon the device. Specify a key for
keyboards.

**Returns:**
True if the unbind was successful, false if the device was unknown or description failed.

**Example:**
```cpp
moveMap.unbind("keyboard", "space");
```

```cpp
bool ActionMap::unbindObj(string device, string action, string obj)
```

Remove any object-binding on an input device and action.

**Parameters:**
- **device** The device to bind to. Can be keyboard, mouse, joystick or gamepad.
- **action** The device action to unbind from. The action is dependant upon the device. Specify a key for keyboards.
- **obj** The object to perform unbind against.

**Returns:**
True if the unbind was successful, false if the device was unknown or description failed.

**Example:**
```cpp
moveMap.unbindObj("keyboard", "numpad1", 
```
AdvancedLightBinManager Class Reference

[Lighting]

Rendering Manager responsible for lighting, shadows, and global variables affecting both. More...

Inheritance diagram for AdvancedLightBinManager:

```
SimObject

<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RenderBinManager</td>
</tr>
<tr>
<td>----------------</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RenderTargetBinManager</td>
</tr>
<tr>
<td>----------------</td>
</tr>
</tbody>
</table>

AdvancedLightBinManager
```

List of all members.
Detailed Description

Rendering Manager responsible for lighting, shadows, and global variables affecting both.

Should not be exposed to TorqueScript as a game object, meant for internal use only

Copyright © GarageGames, LLC. All Rights Reserved.
AIClient Class Reference
[AI, Networking]

Simulated client driven by AI commands. More...

Inheritance diagram for AIClient:

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td><code>getAimLocation</code></td>
<td>ai.getAimLocation();</td>
</tr>
<tr>
<td>string</td>
<td><code>getLocation</code></td>
<td>ai.getLocation();</td>
</tr>
<tr>
<td>string</td>
<td><code>getMoveDestination</code></td>
<td>ai.getMoveDestination();</td>
</tr>
<tr>
<td>int</td>
<td><code>getTargetObject</code></td>
<td>ai.getTargetObject();</td>
</tr>
<tr>
<td>void</td>
<td><code>missionCycleCleanup</code></td>
<td>ai.missionCycleCleanup();</td>
</tr>
<tr>
<td>void</td>
<td><code>move</code></td>
<td>ai.move();</td>
</tr>
<tr>
<td>void</td>
<td><code>moveForward</code></td>
<td>ai.moveForward();</td>
</tr>
<tr>
<td>void</td>
<td><code>setAimLocation</code></td>
<td>ai.setAimLocation(x, y, z);</td>
</tr>
<tr>
<td>void</td>
<td><code>setMoveDestination</code></td>
<td>ai.setMoveDestination(x, y, z);</td>
</tr>
<tr>
<td>void</td>
<td><code>setMoveSpeed</code></td>
<td>ai.setMoveSpeed( float );</td>
</tr>
<tr>
<td>void</td>
<td><code>setTargetObject</code></td>
<td>ai.setTargetObject( obj );</td>
</tr>
<tr>
<td>void</td>
<td><code>stop</code></td>
<td>ai.stop();</td>
</tr>
</tbody>
</table>
**Detailed Description**

Simulated client driven by AI commands.

This object is derived from the `AIConnection` class. It introduces its own `Player` object to solidify the purpose of this class: Simulated client connecting as a player.

To get more specific, if you want a strong alternative to `AIPlayer` (and wish to make use of the `AIConnection` structure), consider `AIClient`. `AIClient` inherits from `AIConnection`, contains quite a bit of functionality you will find in `AIPlayer`, and has its own `Player` object.

**Note:**

This is a legacy class, which you are discouraged from using as it will most likely be deprecated in a future version. For now it has been left in for backwards compatibility with TGE and the old RTS Kit. Use `AIPlayer` instead.

**See also:**

`AIPlayer`, `AIConnection`
## Member Data Documentation

<table>
<thead>
<tr>
<th>Member Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>string AIClient::getAimLocation</code></td>
<td>ai.getAimLocation();</td>
</tr>
<tr>
<td><code>string AIClient::getLocation</code></td>
<td>ai.getLocation();</td>
</tr>
<tr>
<td><code>string AIClient::getMoveDestination</code></td>
<td>ai.getMoveDestination();</td>
</tr>
<tr>
<td><code>int AIClient::getTargetObject</code></td>
<td>ai.getTargetObject();</td>
</tr>
<tr>
<td><code>void AIClient::missionCycleCleanup</code></td>
<td>ai.missionCycleCleanup();</td>
</tr>
<tr>
<td><code>void AIClient::move</code></td>
<td>ai.move();</td>
</tr>
<tr>
<td><code>void AIClient::moveForward</code></td>
<td>ai.moveForward();</td>
</tr>
<tr>
<td>Function</td>
<td>Code</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>void <code>AIClient::setAimLocation</code></td>
<td><code>ai.setAimLocation( x y z );</code></td>
</tr>
<tr>
<td>void <code>AIClient::setMoveDestination</code></td>
<td><code>ai.setMoveDestination( x y z );</code></td>
</tr>
<tr>
<td>void <code>AIClient::setMoveSpeed</code></td>
<td><code>ai.setMoveSpeed( float );</code></td>
</tr>
<tr>
<td>void <code>AIClient::setTargetObject</code></td>
<td><code>ai.setTargetObject( obj );</code></td>
</tr>
<tr>
<td>void <code>AIClient::stop</code></td>
<td><code>ai.stop();</code></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
AIConnection Class Reference  
[AI, Networking]

Special client connection driven by an AI, rather than a human.  

More...

Inheritance diagram for AIConnection:

List of all members.
# Public Member Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>getMove</code> (string field)</td>
<td>Get the given field of a move.</td>
</tr>
<tr>
<td>bool</td>
<td><code>getTrigger</code> (int trigger)</td>
<td>Is the given trigger set?</td>
</tr>
<tr>
<td>void</td>
<td><code>setFreeLook</code> (bool isFreeLook)</td>
<td>Enable/disable freelook on the current move.</td>
</tr>
<tr>
<td>void</td>
<td><code>setMove</code> (string field, float value)</td>
<td>Set a field on the current move.</td>
</tr>
<tr>
<td>void</td>
<td><code>setTrigger</code> (int trigger, bool set)</td>
<td>Set a trigger.</td>
</tr>
</tbody>
</table>
Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td><code>getAddress</code></td>
<td></td>
</tr>
<tr>
<td>bool</td>
<td><code>getFreeLook</code></td>
<td>Is freelook on for the current move?</td>
</tr>
</tbody>
</table>

`getFreeLook()`
Detailed Description

Special client connection driven by an AI, rather than a human.

Unlike other net connections, AIConnection is intended to run unmanned. Rather than gathering input from a human using a device, move events, triggers, and look events are driven through functions like AIConnection::setMove.

In addition to having its own set of functions for managing client move events, a member variable inherited by GameConnection is toggle: mAIControlled. This is useful for a server to determine if a connection is AI driven via the function GameConnection::isAIControlled

AIConnection is an alternative to manually creating an AI driven game object. When you want the server to manage AI, you will create a specific one from script using a class like AIPlayer. If you do not want the server managing the AI and wish to simulate a complete client connection, you will use AIConnection.

To get more specific, if you want a strong alternative to AIPlayer (and wish to make use of the AIConnection structure), consider AIClient. AIClient inherits from AIConnection, contains quite a bit of functionality you will find in AIPlayer, and has its own Player object.

Example:

```cpp
// Create a new AI client connection
%botConnection = aiConnect("MasterBlaster")

// In another area of the code, you can look for using the isAIControlled function
for(%i = 0; %i < ClientGroup.getCount(); %i++)
{
    %client = ClientGroup.getObject(%i);
    if(%client.isAIControlled())
    {
```
// React to this AI controlled client
}
}

Note:
This is a legacy class, which you are discouraged from using as it will most likely be deprecated in a future version. For now it has been left in for backwards compatibility with TGE and the old RTS Kit. Use GameConnection and AIPlayer instead.

See also:
GameConnection, NetConnection, AIclient
### Member Function Documentation

<table>
<thead>
<tr>
<th>Function</th>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>float AIConnection::getMove(string field)</code></td>
<td><code>field</code> One of {'x', 'y', 'z', 'yaw', 'pitch', 'roll'}.</td>
<td>Get the given field of a move.</td>
</tr>
<tr>
<td><code>bool AIConnection::getTrigger(int trigger)</code></td>
<td></td>
<td>Is the given trigger set?</td>
</tr>
<tr>
<td><code>void AIConnection::setFreeLook(bool isFreeLook)</code></td>
<td></td>
<td>Enable/disble freelook on the current move.</td>
</tr>
<tr>
<td><code>void AIConnection::setMove(string field, float value)</code></td>
<td><code>field</code> One of {'x', 'y', 'z', 'yaw', 'pitch', 'roll'}. <code>value</code> Value to set field to.</td>
<td>Set a field on the current move.</td>
</tr>
<tr>
<td><code>void AIConnection::setTrigger(int trigger)</code></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Set a trigger.
## Member Data Documentation

<table>
<thead>
<tr>
<th>Member</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>string AIConnection::getAddress</code></td>
<td></td>
</tr>
<tr>
<td><code>bool AIConnection::getFreeLook</code></td>
<td><code>getFreeLook()</code> Is freelook on for the current move?</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
AIPlayer Class Reference
[AI, Game Objects]

A **Player** object not controlled by conventional input, but by an AI engine. *More...*

Inheritance diagram for AIPlayer:

```
SimObject
    ↓
NetObject
    ↓
SceneObject
    ↓
GameBase
    ↓
ShapeBase
    ↓
Player
    ↓
AIPlayer
```

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td><code>clearAim()</code></td>
<td>Use this to stop aiming at an object or a point.</td>
</tr>
<tr>
<td>Point3F</td>
<td><code>getAimLocation()</code></td>
<td>Returns the point the AIPlayer is aiming at.</td>
</tr>
<tr>
<td>int</td>
<td><code>getAimObject()</code></td>
<td>Gets the object the AIPlayer is targeting.</td>
</tr>
<tr>
<td>Point3F</td>
<td><code>getMoveDestination()</code></td>
<td>Get the AIPlayer's current destination.</td>
</tr>
<tr>
<td>float</td>
<td><code>getMoveSpeed()</code></td>
<td>Gets the move speed of an AI object.</td>
</tr>
<tr>
<td>void</td>
<td><code>setAimLocation(Point3F target)</code></td>
<td>Tells the AIPlayer to aim at the location provided.</td>
</tr>
<tr>
<td>void</td>
<td><code>setAimObject(GameBase targetObject, Point3F offset)</code></td>
<td>Sets the AIPlayer's target object. May optionally set an offset from target location.</td>
</tr>
<tr>
<td>void</td>
<td><code>setMoveDestination(Point3F goal, bool slowDown=true)</code></td>
<td>Tells the AI to move to the location provided.</td>
</tr>
<tr>
<td>void</td>
<td><code>setMoveSpeed(float speed)</code></td>
<td>Sets the move speed for an AI object.</td>
</tr>
<tr>
<td>void</td>
<td><code>stop()</code></td>
<td>Tells the AIPlayer to stop moving.</td>
</tr>
</tbody>
</table>
## Public Attributes

**AI**

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>mMoveTolerance</code></td>
<td>Distance from destination before stopping.</td>
</tr>
<tr>
<td>int</td>
<td><code>moveStuckTestDelay</code></td>
<td>The number of ticks to wait before testing if the AIPlayer is stuck.</td>
</tr>
<tr>
<td>float</td>
<td><code>moveStuckTolerance</code></td>
<td>Distance tolerance on stuck check.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

A **Player** object not controlled by conventional input, but by an AI engine.

The **AIPlayer** provides a **Player** object that may be controlled from script. You control where the player moves and how fast. You may also set where the **AIPlayer** is aiming at -- either a location or another game object.

The **AIPlayer** class does not have a datablock of its own. It makes use of the **PlayerData** datablock to define how it looks, etc. As the **AIPlayer** is an extension of the **Player** class it can mount objects and fire weapons, or mount vehicles and drive them.

While the **PlayerData** datablock is used, there are a number of additional callbacks that are implemented by **AIPlayer** on the datablock. These are listed here:

```c
void onReachDestination(AIPlayer obj) Called when the player has reached its set destination using the setMoveDestination() method. The actual point at which this callback is called is when the AIPlayer is within the mMoveTolerance of the defined destination.

void onMoveStuck(AIPlayer obj) While in motion, if an AIPlayer has moved less than moveStuckTolerance within a single tick, this callback is called. From here you could choose an alternate destination to get the AIPlayer moving again.

void onTargetEnterLOS(AIPlayer obj) When an object is being aimed at (following a call to setAimObject()) and the targeted object enters the AIPlayer's line of sight, this callback is called. The LOS test is a ray from the AIPlayer's eye position to the center of the target's bounding box. The LOS ray test only checks against interiors, statis shapes, and terrain.

void onTargetExitLOS(AIPlayer obj) When an object is being aimed at (following a call to setAimObject()) and the targeted object leaves the AIPlayer's line of sight, this callback is called. The LOS test is a
ray from the AIPlayer's eye position to the center of the target's bounding box. The LOS ray test only checks against interiors, static shapes, and terrain.

**Example:**

```plaintext
// Create the demo player object
%player = new AiPlayer()
{
    dataBlock = DemoPlayer;
    path = "";
};
```

**See also:**

[Player](#) for a list of all inherited functions, variables, and base description
Member Function Documentation

void AIPlayer::clearAim()

Use this to stop aiming at an object or a point.

See also:
    setAimLocation()
    setAimObject()

Point3F AIPlayer::getAimLocation()

Returns the point the AIPlayer is aiming at.

This will reflect the position set by setAimLocation(), or the position of the object that the bot is now aiming at. If the bot is not aiming at anything, this value will change to whatever point the bot's current line-of-sight intercepts.

Returns:
    World space coordinates of the object AI is aiming at. Formatted as "X Y Z".

See also:
    setAimLocation()
    setAimObject()

int AIPlayer::getAimObject()

Gets the object the AIPlayer is targeting.

Returns:
    Returns -1 if no object is being aimed at, or the SimObjectID of the object the AIPlayer is aiming at.
See also:
   setAimObject()  

Point3F AIPlayer::getMoveDestination()  

Get the AIPlayer's current destination.

Returns:
   Returns a point containing the "x y z" position of the AIPlayer's current move destination. If no move destination has yet been set, this returns "0 0 0".

See also:
   setMoveDestination()  

float AIPlayer::getMoveSpeed()  

Gets the move speed of an AI object.

Returns:
   A speed multiplier between 0.0 and 1.0.

See also:
   setMoveSpeed()  

void AIPlayer::setAimLocation(Point3F target)  

Tells the AIPlayer to aim at the location provided.

Parameters:
   target An "x y z" position in the game world to target.

See also:
void AIPlayer::setAimObject(GameBase targetObject, Point3F offset)

Sets the AIPlayer's target object. May optionally set an offset from target location.

Parameters:

- targetObject: The object to target
- offset: Optional three-element offset vector which will be added to the position of the aim object.

Example:

```
// Without an offset
%ai.setAimObject(%target);

// With an offset
// Cause our AI object to aim at the target offset (0, 0, 1) so you don't aim at the target's feet
%ai.setAimObject(%target, "0 0 1");
```

See also:
- getAimLocation()
- getAimObject()
- clearAim()
Parameters:

\[
\text{goal} \quad \text{Coordinates in world space representing location to move to.}
\]

A boolean value. If set to true, the bot will slow down when it gets within 5-meters of its move destination. If false, the bot will stop abruptly when it reaches the move destination. By default, this is true.

Note:

Upon reaching a move destination, the bot will clear its move destination and calls to getMoveDestination will return "0 0 0".

See also:

getMoveDestination()

---

**void AIPlayer::setMoveSpeed(float speed)**

Sets the move speed for an AI object.

**Parameters:**

A speed multiplier between 0.0 and 1.0. This is \textit{speed} multiplied by the AIPlayer's base movement rates (as defined in its PlayerData datablock)

See also:

getMoveDestination()

---

**void AIPlayer::stop( )**

Tells the AIPlayer to stop moving.
Member Data Documentation

**float AIPlayer::mMoveTolerance**

Distance from destination before stopping.

When the AIPlayer is moving to a given destination it will move to within this distance of the destination and then stop. By providing this tolerance it helps the AIPlayer from never reaching its destination due to minor obstacles, rounding errors on its position calculation, etc. By default it is set to 0.25.

**int AIPlayer::moveStuckTestDelay**

The number of ticks to wait before testing if the AIPlayer is stuck.

When the AIPlayer is asked to move, this property is the number of ticks to wait before the AIPlayer starts to check if it is stuck. This delay allows the AIPlayer to accelerate to full speed without its initial slow start being considered as stuck.

**Note:**

Set to zero to have the stuck test start immediately.

**float AIPlayer::moveStuckTolerance**

Distance tolerance on stuck check.

When the AIPlayer is moving to a given destination, if it ever moves less than this tolerance during a single tick, the AIPlayer is considered stuck. At this point the onMoveStuck() callback is called on the datablock.
Provides an AI controlled turret. More...

Inheritance diagram for AITurretShape:

List of all members.
# Public Member Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td><code>activateTurret ()</code></td>
<td>Activate a turret from a deactivate state.</td>
</tr>
<tr>
<td>void</td>
<td><code>addToIgnoreList (ShapeBase obj)</code></td>
<td>Adds object to the turret's ignore list.</td>
</tr>
<tr>
<td>void</td>
<td><code>deactivateTurret ()</code></td>
<td>Deactivate a turret from an active state.</td>
</tr>
<tr>
<td>SimObject</td>
<td><code>getTarget ()</code></td>
<td>Get the turret's current target.</td>
</tr>
<tr>
<td>float</td>
<td><code>getWeaponLeadVelocity ()</code></td>
<td>Get the turret's defined projectile velocity that helps with target leading.</td>
</tr>
<tr>
<td>bool</td>
<td><code>hasTarget ()</code></td>
<td>Indicates if the turret has a target.</td>
</tr>
<tr>
<td>void</td>
<td><code>recenterTurret ()</code></td>
<td>Recenter the turret's weapon.</td>
</tr>
<tr>
<td>void</td>
<td><code>removeFromIgnoreList (ShapeBase obj)</code></td>
<td>Removes object from the turret's ignore list.</td>
</tr>
<tr>
<td>void</td>
<td><code>resetTarget ()</code></td>
<td>Resets the turret's target tracking.</td>
</tr>
<tr>
<td>void</td>
<td><code>setAllGunsFiring (bool fire)</code></td>
<td>Set the firing state of the turret's guns.</td>
</tr>
<tr>
<td>void</td>
<td><code>setGunSlotFiring (int slot, bool fire)</code></td>
<td>Set the firing state of the given gun slot.</td>
</tr>
<tr>
<td>void</td>
<td><code>setTurretState (string newState, bool force=false)</code></td>
<td>Set the turret's current state.</td>
</tr>
<tr>
<td>void</td>
<td><code>setWeaponLeadVelocity (float velocity)</code></td>
<td>Set the turret's projectile velocity to help lead the target.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>void <strong>startScanForTargets</strong> ()</td>
<td>Begin scanning for a target.</td>
<td></td>
</tr>
<tr>
<td>void <strong>startTrackingTarget</strong> ()</td>
<td>Have the turret track the current target.</td>
<td></td>
</tr>
<tr>
<td>void <strong>stopScanForTargets</strong> ()</td>
<td>Stop scanning for targets.</td>
<td></td>
</tr>
<tr>
<td>void <strong>stopTrackingTarget</strong> ()</td>
<td>Stop the turret from tracking the current target.</td>
<td></td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td><code>isRenderable</code></td>
<td>Enables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td><code>isSelectable</code></td>
<td>Enables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Provides an AI controlled turret.

Uses the **AITurretShapeData** datablock, which is based on the **TurretShapeData** datablock for common properties.

**AITurretShape** builds an AI controlled turret. It uses a state machine and properties as defined in **AITurretShapeData** to decide how to find targets and what to do with them. As with **TurretShape** (which **AITurretShape** derives from) the **AITurretShape** class provides the base on which **ShapeBaseImageData** weapons may be mounted.
Overview

The **AITurretShape** functions through the use of a state machine as defined in its **AITurretShapeData** datablock. It is very similar to how **ShapeBaselImageData** works. This allows you to customize exactly how the turret behaves while it searches for a target, and what it does once it has a target. But in general, the AI turret goes through a number of stages:

**Scanning**

The AI turret usually starts off by scanning for a suitable target. This is done by checking for targets within a pie wedge shaped volume in front of the turret based on where the scanPoint node is placed. The turret takes cover into account when searching for a target so that it doesn't "cheat".

**Gained Target**

Once a target is acquired the turret attempts to follow it. Usually at this point the turret activates its weapon. If a target is lost due to it going behind cover, the turret will attempt to follow and reacquire the target using its last known position and velocity. The amount of time allowed for this attempt is defined by **AITurretShapeData::trackLostTargetTime**.

**Lost Target**

If the target is lost (either by going behind cover or it is dead) the turret returns to its scanning mode to find another victim.

**Destroyed**

If the AI turret is destroyed then it can go into a special state to show the user that it has been destroyed. As with **TurretShape** turrets a **AITurretShape** may respawn after a set amount of time (see **TurretShape** and **TurretShape::doRespawn()**).

**Deployable Turret**
In addition to AI turrets being placed within a mission, it is also possible for a player to deploy a turret such as throwing one from their inventory. When a turret has been tossed it will be in a *Thrown* state and usually in an inactive mode. Once the turret comes to rest on the ground it moves into a *Deploy* state where it may unfold to get ready. Once ready the turret begins the scanning process. As the AI turret's state machine may be customized for your specific circumstances, the way in which turrets are deployed by a player is up to you. An AI turret could be thrown in a fully working state, ready to take out targets before the turret even hits the ground.
Example State Machine

Here is an example AITurretShapeData datablock with a defined state machine and the script to support the state machine. This is just one possible example.

Example:

```plaintext
// AI Turret
//

datablock AITurretShapeData(AITurret) {
    category = "Turrets";
    shapeFile = "art/shapes/weapons/Turret/

    maxDamage = 70;
    destroyedLevel = 70;
    explosion = GrenadeExplosion;

    simpleServerCollision = false;

    zRotOnly = false;

    // Rotation settings
    minPitch = 15;
    maxPitch = 80;
    maxHeading = 90;
    headingRate = 50;
    pitchRate = 50;

    // Scan settings
    maxScanPitch = 10;
}
maxScanHeading = 30;
maxScanDistance = 20;
trackLostTargetTime = 2;

maxWeaponRange = 30;

weaponLeadVelocity = 0;

// Weapon mounting
numWeaponMountPoints = 1;

weapon[0] = AITurretHead;
weaponAmmo[0] = AITurretAmmo;
weaponAmmoAmount[0] = 10000;

maxInv[AITurretHead] = 1;
maxInv[AITurretAmmo] = 10000;

// Initial start up state
stateName[0] = "Preactivate"
stateTransitionOnAtRest[0] = "Scanning"
stateTransitionOnNotAtRest[0] = "Thrown"

// Scan for targets
stateName[1] = "Scanning"
stateScan[1] = true
stateTransitionOnTarget[1] = "Target"
stateSequence[1] = "scan"
stateScript[1] = "OnScanning"

// Have a target
stateName[2] = "Target"
stateTransitionOnNoTarget[2] = "NoTarget"
stateTransitionOnTimeout[2] = "Firing"
stateTimeoutValue[2] = 2.0;
stateScript[2] = "OnTarget"

// Fire at target
stateName[3] = "Firing"
stateFire[3] = true
stateTransitionOnNoTarget[3] = "NoTarget"
stateScript[3] = "OnFiring"

// Lost target
stateName[4] = "NoTarget"
stateTimeoutValue[4] = 2.0;
stateScript[4] = "OnNoTarget"

// Player thrown turret
stateName[5] = "Thrown"
stateTransitionOnAtRest[5] = "Deploy"
stateSequence[5] = "throw"
stateScript[5] = "OnThrown"

// Player thrown turret is deploying
stateName[6] = "Deploy"
stateTimeoutValue[6] = 2.5;
stateSequence[6] = "deploy"
stateScaleAnimation[6] = true
stateScript[6] = "OnDeploy"

// Special state that is set when the turret is destroyed.
// This state is set in the onDestroyed() callback.
stateName[7] = "Destroyed"
stateSequence[7] = "destroyed";

// Deployable AI Turret
datablock AITurretShapeData(DeployableTurret)
{
    // Mission editor category
    category = "Weapon";

    className = "DeployableTurretWeapon";

    startLoaded = false;

    // Basic Item properties
    mass = 1.5;
    elasticity = 0.1;
    friction = 0.6;
    simpleServerCollision = false;

    // Dynamic properties defined by the scripts
    PreviewImage = 'turret.png';
    pickUpName = "a deployable turret";
    description = "Deployable Turret";
    image = DeployableTurretImage;
    reticle = "blank";
    zoomReticle = 'blank';
};
function AITurretShapeData::onAdd(%this, %obj)
{
    Parent::onAdd(%this, %obj);

    %obj.mountable = false;
}

// Player has thrown a deployable turret.
function AITurretShapeData::onThrow(%this, %user, %amount)
{
    // Remove the object from the inventory
    if (%amount $= "")
        %amount = 1;
    if (%this.maxInventory !$= "")
        if (%amount > %this.maxInventory)
            %amount = %this.maxInventory;
    if (!%amount)
        return 0;
    %user.decInventory(%this, %amount);

    // Construct the actual object in the world, in the mission group so it's cleaned up when done. The turret's rotation matches the player's.
    %rot = %user.getEulerRotation();
    %obj = new AITurretShape()
    {
        datablock = %this;
        rotation = "0 0 1 " @ getWord(%rot, 2); 
        count = 1;
        sourceObject = %user;
        client = %user.client;
        isAiControlled = true;
MissionGroup.add(%obj);

// Let the turret know that we're a friend
%obj.addToIgnoreList(%user);

// We need to add this turret to a list
// the turret will still ignore our play
%client = %user.client;
if (%client)
{
  if (!%client.ownedTurrets)
  {
    %client.ownedTurrets = new SimSet
  }

  // Go through the client's owned turret
  // a friend of every turret and every
  // Commence hugging!
  for (%i=0; %i<%client.ownedTurrets.getCount(); %i++)
  {
    %turret = %client.ownedTurrets.getObject(%i);
    %turret.addToIgnoreList(%obj);
    %obj.addToIgnoreList(%turret);
  }

  // Add ourselves to the client's own
  %client.ownedTurrets.add(%obj);
}

return %obj;
function AITurretShapeData::onDestroyed(%turret)
{
    // This method is invoked by the ShapeBase code whenever the object's damage state changes.
    %turret.playAudio(0, TurretDestroyed);
    %turret.setAllGunsFiring(false);
    %turret.resetTarget();
    %turret.setTurretState( "Destroyed", true );

    // Set the weapons to destroyed
    for(%i = 0; %i < %this.numWeaponMountPoints; %i++)
    {
        %turret.setImageGenericTrigger(%i, 0);
    }

    Parent::onDestroyed(%this, %turret, %lastState);
}

function AITurretShapeData::OnScanning(%this)
{
    //echo("AITurretShapeData::OnScanning:

    %turret.startScanForTargets();
    %turret.playAudio(0, TurretScanningSound);
}

function AITurretShapeData::OnTarget(%this, %turret)
{
    //echo("AITurretShapeData::OnTarget: ");

    %turret.startTrackingTarget();
    %turret.playAudio(0, TargetAquiredSound);
function AITurretShapeData::OnNoTarget(%this)
{
    //echo("AITurretShapeData::OnNoTarget:");
    %turret.setAllGunsFiring(false);
    %turret.recenterTurret();
    %turret.playAudio(0, TargetLostSound);
}

function AITurretShapeData::OnFiring(%this, %turret)
{
    //echo("AITurretShapeData::OnFiring: ");
    %turret.setAllGunsFiring(true);
}

function AITurretShapeData::OnThrown(%this, %turret)
{
    //echo("AITurretShapeData::OnThrown: ");
    %turret.playAudio(0, TurretThrown);
}

function AITurretShapeData::OnDeploy(%this)
{
    //echo("AITurretShapeData::OnDeploy: ");
    // Set the weapons to loaded
    for(%i = 0; %i < %this.numWeaponMountPoints; %i++)
    {
        %turret.setImageLoaded(%i, true);
    }
}
And here is the above example state machine's flow:
Shape File Nodes

In addition to the required TurretBase nodes, AITurretShape makes use of additional nodes within the shape file to allow the AI to do its work. The first is the 'scanPoint' node. This is used by the AI to project a pie wedge shaped scanning volume in which to detect possible targets. The scanPoint node is at the apex of the scanning wedge. If the scanPoint node is not present within the shape file then the turret's world transform is used.

The second is the 'aimPoint' node. Once the AI turret has obtained a target the aimPoint is used to point the turret at the target. Specifically, the turret rotates in both pitch and heading such that the aimPoint points at the target. If you're using a weapon that doesn't have its muzzle point on the same plane as its mount point (known as an off-axis weapon) then be sure to place the aimPoint at a z position equivalent to the weapon's muzzle point. This allows for the correct pitch calculation. If the aimPoint is not found on the turret's shape, then the pitch node will be used.
Ignore List

AI turrets keep track of an ignore list. This is used by default to stop a player deployed turret from targeting its owner, even when that owner is killed and respawns. But this ignore list could also be used to have the turret ignore team mates, squad members, invisible players, etc. Use `AI.TurretShape::addToIgnoreList()` and `AI.TurretShape::removeFromIgnoreList()` to manipulate this list. You should also look in scripts/server/turret.cs at `AI.TurretShapeData::onThrow()` to see how the ignore list is handled and deployed turrets are kept track of on a per connected client basis.

See also:

- `AI.TurretShapeData`
- `TurretShape`
- `TurretShapeData`
- `ShapeBaseImageData`
Member Function Documentation

void AITurretShape::activateTurret()

Activate a turret from a deactivate state.

void AITurretShape::addToIgnoreList(ShapeBase obj)

Adds object to the turret's ignore list.

All objects in this list will be ignored by the turret's targeting.

Parameters:

  - obj The ShapeBase object to ignore.

void AITurretShape::deactivateTurret()

Deactivate a turret from an active state.

SimObject AITurretShape::getTarget()

Get the turret's current target.

Returns:

  The object that is the target's current target, or 0 if no target.

float AITurretShape::getWeaponLeadVelocity()

Get the turret's defined projectile velocity that helps with target leading.
Returns:
The defined weapon projectile speed, or 0 if leading is disabled.

bool AITurretShape::hasTarget(
)

Indicates if the turret has a target.

Returns:
True if the turret has a target.

void AITurretShape::recenterTurret(
)

Recenter the turret's weapon.

void AITurretShape::removeFromIgnoreList(ShapeBase obj)

Removes object from the turret's ignore list.
All objects in this list will be ignored by the turret's targeting.

Parameters:
obj The ShapeBase object to once again allow for targeting.

void AITurretShape::resetTarget(
)

Resets the turret's target tracking.
Only resets the internal target tracking. Does not modify the turret's facing.
void AITurretShape::setAllGunsFiring (bool fire )

Set the firing state of the turret's guns.

**Parameters:**

*fire* Set to true to activate all guns. False to deactivate them.

void AITurretShape::setGunSlotFiring (int slot,
                                          bool fire )

Set the firing state of the given gun slot.

**Parameters:**

*slot* The gun to modify. Valid range is 0-3 that corresponds to the weapon mount point.

*fire* Set to true to activate the gun. False to deactivate it.

void AITurretShape::setTurretState (string newState,
                                           bool force = false)

Set the turret's current state.

Normally the turret's state comes from updating the state machine but this method allows you to override this and jump to the requested state immediately.

**Parameters:**

*newState* The name of the new state.

*force* Is true then force the full processing of the new state even if it is the same as the current state. If false then only the time out value is reset and the state’s script method is called, if any.
void AITurretShape::setWeaponLeadVelocity(float velocity)

Set the turret's projectile velocity to help lead the target.

This value normally comes from AITurretShapeData::weaponLeadVelocity but this method allows you to override the datablock value. This can be useful if the turret changes ammunition, uses a different weapon than the default, is damaged, etc.

**Note:**

Setting this to 0 will disable target leading.

void AITurretShape::startScanForTargets(

Begin scanning for a target.

void AITurretShape::startTrackingTarget(

Have the turret track the current target.

void AITurretShape::stopScanForTargets(

Stop scanning for targets.

**Note:**

Only impacts the scanning for new targets. Does not effect a turret's current target lock.

void AITurretShape::stopTrackingTarget( )
Stop the turret from tracking the current target.
AILturretShapeData Class Reference
[Game Objects]

Defines properties for an AITurretShape object. More...

Inheritance diagram for AITurretShapeData:

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>maxScanDistance</code></td>
<td>Maximum distance to scan.</td>
</tr>
<tr>
<td>float</td>
<td><code>maxScanHeading</code></td>
<td>Maximum number of degrees to scan left and right.</td>
</tr>
<tr>
<td>float</td>
<td><code>maxScanPitch</code></td>
<td>Maximum number of degrees to scan up and down.</td>
</tr>
<tr>
<td>float</td>
<td><code>maxWeaponRange</code></td>
<td>Maximum distance that the weapon will fire upon a target.</td>
</tr>
<tr>
<td>int</td>
<td><code>scanTickFrequency</code></td>
<td>How often should we perform a full scan when looking for a target.</td>
</tr>
<tr>
<td>int</td>
<td><code>scanTickFrequencyVariance</code></td>
<td>Random amount that should be added to the scan tick frequency each scan period.</td>
</tr>
<tr>
<td>bool</td>
<td><code>stateDirection</code> [31]</td>
<td>Direction of the animation to play in this state.</td>
</tr>
<tr>
<td>bool</td>
<td><code>stateFire</code> [31]</td>
<td>The first state with this set to true is the state entered by the client when it receives the 'fire' event.</td>
</tr>
<tr>
<td>caseString</td>
<td><code>stateName</code> [31]</td>
<td>Name of this state.</td>
</tr>
<tr>
<td>bool</td>
<td><code>stateScaleAnimation</code> [31]</td>
<td>If true, the timeScale of the stateSequence animation will be adjusted such that the sequence plays for stateTimeoutValue seconds.</td>
</tr>
<tr>
<td>bool</td>
<td><code>stateScan</code> [31]</td>
<td>Indicates the turret should perform a continuous scan looking for targets.</td>
</tr>
<tr>
<td>caseString</td>
<td><code>stateScript</code> [31]</td>
<td></td>
</tr>
<tr>
<td>Method to execute on entering this state.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>string</strong> stateSequence [31]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name of the sequence to play on entry to this state.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>float</strong> stateTimeoutValue [31]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time in seconds to wait before transitioning to stateTransitionOnTimeout.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>string</strong> stateTransitionOnActivated [31]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name of the state to transition to when the turret goes from deactivated to activated.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>string</strong> stateTransitionOnAtRest [31]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name of the state to transition to when the turret is at rest (static).</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>string</strong> stateTransitionOnDeactivated [31]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name of the state to transition to when the turret goes from activated to deactivated.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>string</strong> stateTransitionOnNoTarget [31]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name of the state to transition to when the turret loses a target.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>string</strong> stateTransitionOnNotAtRest [31]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name of the state to transition to when the turret is not at rest (not static).</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>string</strong> stateTransitionOnTarget [31]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name of the state to transition to when the turret gains a target.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>string</strong> stateTransitionOnTimeout [31]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name of the state to transition to when we have been in this state for stateTimeoutValue seconds.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>bool</strong> stateWaitForTimeout [31]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>If false, this state ignores stateTimeoutValue and transitions immediately if other transition conditions are met.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>float</strong> trackLostTargetTime</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
How long after the turret has lost the target should it still track it.

<table>
<thead>
<tr>
<th>float</th>
<th>weaponLeadVelocity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Velocity used to lead target.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Defines properties for an *AlTurretShape* object.

See also:

- *AlTurretShape*
- *TurretShapeData*
### Member Data Documentation

**float AITurretShapeData::maxScanDistance**

Maximum distance to scan.

When combined with `maxScanHeading` and `maxScanPitch` this forms a 3D scanning wedge used to initially locate a target.

**float AITurretShapeData::maxScanHeading**

Maximum number of degrees to scan left and right.

**Note:**

Maximum scan heading is 90 degrees.

**float AITurretShapeData::maxScanPitch**

Maximum number of degrees to scan up and down.

**Note:**

Maximum scan pitch is 90 degrees.

**float AITurretShapeData::maxWeaponRange**

Maximum distance that the weapon will fire upon a target.

**int AITurretShapeData::scanTickFrequency**

How often should we perform a full scan when looking for a target. Expressed as the number of ticks between full scans, but no less
than 1.

`int AITurretShapeData::scanTickFrequencyVariance`

Random amount that should be added to the scan tick frequency each scan period.

Expressed as the number of ticks to randomly add, but no less than zero.

`bool AITurretShapeData::stateDirection[31]`

Direction of the animation to play in this state.

True is forward, false is backward.

`bool AITurretShapeData::stateFire[31]`

The first state with this set to true is the state entered by the client when it receives the 'fire' event.

`caseString AITurretShapeData::stateName[31]`

Name of this state.

`bool AITurretShapeData::stateScaleAnimation[31]`

If true, the timeScale of the stateSequence animation will be adjusted such that the sequence plays for stateTimeoutValue seconds.
bool *AITurretShapeData::stateScan*

Indicates the turret should perform a continuous scan looking for targets.

**caseString *AITurretShapeData::stateScript***

Method to execute on entering this state.

Scoped to *AITurretShapeData*.

**string *AITurretShapeData::stateSequence***

Name of the sequence to play on entry to this state.

**float *AITurretShapeData::stateTimeoutValue***

Time in seconds to wait before transitioning to *stateTransitionOnTimeout*.

**string *AITurretShapeData::stateTransitionOnActivated***

Name of the state to transition to when the turret goes from deactivated to activated.

**string *AITurretShapeData::stateTransitionOnAtRest***

Name of the state to transition to when the turret is at rest (static).

**string *AITurretShapeData::stateTransitionOnDeactivated***
Name of the state to transition to when the turret goes from activated to deactivated.

**string AITurretShapeData::stateTransitionOnNoTarget[31]**

Name of the state to transition to when the turret loses a target.

**string AITurretShapeData::stateTransitionOnNotAtRest[31]**

Name of the state to transition to when the turret is not at rest (not static).

**string AITurretShapeData::stateTransitionOnTarget[31]**

Name of the state to transition to when the turret gains a target.

**string AITurretShapeData::stateTransitionOnTimeout[31]**

Name of the state to transition to when we have been in this state for stateTimeoutValue seconds.

**bool AITurretShapeData::stateWaitForTimeout[31]**

If false, this state ignores stateTimeoutValue and transitions immediately if other transition conditions are met.

**float AITurretShapeData::trackLostTargetTime**

How long after the turret has lost the target should it still track it.
Expressed in seconds.

float AITurretShapeData::weaponLeadVelocity

Velocity used to lead target.

If value \(\leq 0\), don't lead target.
ArrayObject Class Reference

[Scripting]

Data structure for storing indexed sequences of key/value pairs.
More...

Inheritance diagram for ArrayObject:

```
  SimObject
   ^
  ArrayObject
```

[legend]

List of all members.
Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void add (string key, string value=&quot;&quot;)</code></td>
<td>Adds a new element to the end of an array (same as <code>push_back()</code>).</td>
</tr>
<tr>
<td><code>bool append (ArrayObject target)</code></td>
<td>Appends the target array to the array object.</td>
</tr>
<tr>
<td><code>int count ()</code></td>
<td>Get the number of elements in the array.</td>
</tr>
<tr>
<td><code>int countKey (string key)</code></td>
<td>Get the number of times a particular key is found in the array.</td>
</tr>
<tr>
<td><code>int countValue (string value)</code></td>
<td>Get the number of times a particular value is found in the array.</td>
</tr>
<tr>
<td><code>bool crop (ArrayObject target)</code></td>
<td>Removes elements with matching keys from array.</td>
</tr>
<tr>
<td><code>bool duplicate (ArrayObject target)</code></td>
<td>Alters array into an exact duplicate of the target array.</td>
</tr>
<tr>
<td><code>void echo ()</code></td>
<td>Echos the array contents to the console.</td>
</tr>
<tr>
<td><code>void empty ()</code></td>
<td>Emptys all elements from an array.</td>
</tr>
<tr>
<td><code>void erase (int index)</code></td>
<td>Removes an element at a specific position from the array.</td>
</tr>
<tr>
<td><code>int getCurrent ()</code></td>
<td>Gets the current pointer index.</td>
</tr>
<tr>
<td><code>int getIndexFromKey (string key)</code></td>
<td>Search the array from the current position for the key.</td>
</tr>
<tr>
<td><code>int getIndexFromValue (string value)</code></td>
<td>Search the array from the current position for the element.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------------------------------------------------</td>
</tr>
<tr>
<td>string <strong>getKey</strong> (int index)</td>
<td>Get the key of the array element at the submitted index.</td>
</tr>
<tr>
<td>string <strong>getValue</strong> (int index)</td>
<td>Get the value of the array element at the submitted index.</td>
</tr>
<tr>
<td>void <strong>insert</strong> (string key, string value, int index)</td>
<td>Adds a new element to a specified position in the array.</td>
</tr>
<tr>
<td>int <strong>moveFirst</strong> ()</td>
<td>Moves array pointer to start of array.</td>
</tr>
<tr>
<td>int <strong>moveLast</strong> ()</td>
<td>Moves array pointer to end of array.</td>
</tr>
<tr>
<td>int <strong>moveNext</strong> ()</td>
<td>Moves array pointer to next position.</td>
</tr>
<tr>
<td>int <strong>movePrev</strong> ()</td>
<td>Moves array pointer to prev position.</td>
</tr>
<tr>
<td>void <strong>pop_back</strong> ()</td>
<td>Removes the last element from the array.</td>
</tr>
<tr>
<td>void <strong>pop_front</strong> ()</td>
<td>Removes the first element from the array.</td>
</tr>
<tr>
<td>void <strong>push_back</strong> (string key, string value=&quot;&quot;)</td>
<td>Adds a new element to the end of an array.</td>
</tr>
<tr>
<td>void <strong>push_front</strong> (string key, string value=&quot;&quot;)</td>
<td>Adds a new element to the front of an array.</td>
</tr>
<tr>
<td>void <strong>setCurrent</strong> (int index)</td>
<td>Sets the current pointer index.</td>
</tr>
<tr>
<td>void <strong>setKey</strong> (string key, int index)</td>
<td>Set the key at the given index.</td>
</tr>
<tr>
<td>void <strong>setValue</strong> (string value, int index)</td>
<td>Set the value at the given index.</td>
</tr>
<tr>
<td>void <strong>sort</strong> (bool ascending=false)</td>
<td>Alpha sorts the array by value.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>sorta()</code></td>
<td>Alpha sorts the array by value in ascending order.</td>
</tr>
<tr>
<td><code>sortd()</code></td>
<td>Alpha sorts the array by value in descending order.</td>
</tr>
<tr>
<td><code>sortf()</code></td>
<td>Sorts the array by value in ascending order using the given callback function.</td>
</tr>
<tr>
<td><code>sortfd()</code></td>
<td>Sorts the array by value in descending order using the given callback function.</td>
</tr>
<tr>
<td><code>sortfk()</code></td>
<td>Sorts the array by key in ascending order using the given callback function.</td>
</tr>
<tr>
<td><code>sortkd()</code></td>
<td>Sorts the array by key in descending order using the given callback function.</td>
</tr>
<tr>
<td><code>sortk()</code></td>
<td>Alpha sorts the array by key.</td>
</tr>
<tr>
<td><code>sortka()</code></td>
<td>Alpha sorts the array by key in ascending order.</td>
</tr>
<tr>
<td><code>sortkd()</code></td>
<td>Alpha sorts the array by key in descending order.</td>
</tr>
<tr>
<td><code>sortn()</code></td>
<td>Numerically sorts the array by value.</td>
</tr>
<tr>
<td><code>sortna()</code></td>
<td>Numerically sorts the array by value in ascending order.</td>
</tr>
<tr>
<td><code>sortnd()</code></td>
<td>Numerically sorts the array by value in descending order.</td>
</tr>
<tr>
<td><code>sortnk()</code></td>
<td>Numerically sorts the array by key.</td>
</tr>
<tr>
<td><code>sortnka()</code></td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>void <code>sortnkd ()</code></td>
<td>Numerical sorts the array by key in descending order.</td>
</tr>
<tr>
<td>void <code>uniqueKey ()</code></td>
<td>Removes any elements that have duplicated keys (leaving the first instance).</td>
</tr>
<tr>
<td>void <code>uniqueValue ()</code></td>
<td>Removes any elements that have duplicated values (leaving the first instance).</td>
</tr>
</tbody>
</table>
### Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>caseSensitive</td>
<td>Makes the keys and values case-sensitive.</td>
</tr>
<tr>
<td>caseString</td>
<td>key</td>
<td>Helper field which allows you to add new key['keyname'] = value pairs.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Data structure for storing indexed sequences of key/value pairs.

This is a powerful array class providing PHP style arrays in TorqueScript.

The following features are supported:

- **array pointers**: this allows you to move forwards or backwards through the array as if it was a list, including jumping to the start or end.
- **sorting**: the array can be sorted in either alphabetic or numeric mode, on the key or the value, and in ascending or descending order.
- **add/remove elements**: elements can be pushed/popped from the start or end of the array, or can be inserted/erased from anywhere in the middle.
- **removal of duplicates**: remove duplicate keys or duplicate values.
- **searching**: search the array and return the index of a particular key or value.
- **counting**: count the number of instances of a particular value or key in the array, as well as the total number of elements.
- **advanced features**: array append, array crop and array duplicate.

Array element keys and values can be strings or numbers.
Member Function Documentation

void ArrayObject::add (string key,
    string value = "")

Adds a new element to the end of an array (same as push_back()).

**Parameters:**

- *key* Key for the new element
- *value* Value for the new element

bool ArrayObject::append (ArrayObject target)

Appends the target array to the array object.

**Parameters:**

- *target* ArrayObject to append to the end of this array

int ArrayObject::count (

Get the number of elements in the array.

int ArrayObject::countKey (string key)

Get the number of times a particular key is found in the array.

**Parameters:**

- *key* Key value to count
int ArrayObject::countValue(string value)

Get the number of times a particular value is found in the array.

**Parameters:**

value Array element value to count

bool ArrayObject::crop(ArrayObject target)

Removes elements with matching keys from array.

**Parameters:**

target ArrayObject containing keys to remove from this array

bool ArrayObject::duplicate(ArrayObject target)

Alters array into an exact duplicate of the target array.

**Parameters:**

target ArrayObject to duplicate

void ArrayObject::echo()

Echos the array contents to the console.

void ArrayObject::empty()

Emptys all elements from an array.

void ArrayObject::erase(int index)
Removes an element at a specific position from the array.

**Parameters:**

- *index* 0-based index of the element to remove

```cpp
int ArrayObject::getCurrent() {}
```

Gets the current pointer index.

```cpp
int ArrayObject::getIndexFromKey(string key) {}
```

Search the array from the current position for the key.

**Parameters:**

- *value* Array key to search for

**Returns:**

Index of the first element found, or -1 if none

```cpp
int ArrayObject::getIndexFromValue(string value) {}
```

Search the array from the current position for the element.

**Parameters:**

- *value* Array value to search for

**Returns:**

Index of the first element found, or -1 if none

```cpp
string ArrayObject::getKey(int index) {}
```

Get the key of the array element at the submitted index.
Parameters:

- `index` 0-based index of the array element to get

Returns:

The key associated with the array element at the specified index, or "" if the index is out of range.

```cpp
string ArrayObject::getValue(int index)
```

Get the value of the array element at the submitted index.

Parameters:

- `index` 0-based index of the array element to get

Returns:

The value of the array element at the specified index, or "" if the index is out of range.

```cpp
void ArrayObject::insert(string key, string value, int index)
```

Adds a new element to a specified position in the array.

- `index = 0` will insert an element at the start of the array (same as `push_front()`)
- `index = array.count()` will insert an element at the end of the array (same as `push_back()`)

Parameters:

- `key` Key for the new element
- `value` Value for the new element
- `index` 0-based index at which to insert the new element
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ArrayObject::moveFirst()</code></td>
<td>Moves array pointer to start of array.</td>
<td>Returns the new array pointer</td>
</tr>
<tr>
<td><code>ArrayObject::moveLast()</code></td>
<td>Moves array pointer to end of array.</td>
<td>Returns the new array pointer</td>
</tr>
<tr>
<td><code>ArrayObject::moveNext()</code></td>
<td>Moves array pointer to next position.</td>
<td>Returns the new array pointer, or -1 if already at the end</td>
</tr>
<tr>
<td><code>ArrayObject::movePrev()</code></td>
<td>Moves array pointer to prev position.</td>
<td>Returns the new array pointer, or -1 if already at the start</td>
</tr>
<tr>
<td><code>ArrayObject::pop_back()</code></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Removes the last element from the array.

```cpp
void ArrayObject::pop_front()
```

Removes the first element from the array.

```cpp
void ArrayObject::push_back(string key, string value = "")
```

Adds a new element to the end of an array.

**Parameters:**
- `key` Key for the new element
- `value` Value for the new element

```cpp
void ArrayObject::push_front(string key, string value = "")
```

Adds a new element to the front of an array.

```cpp
void ArrayObject::setCurrent(int index)
```

Sets the current pointer index.

**Parameters:**
- `index` New 0-based pointer index

```cpp
void ArrayObject::setKey(string key,
```
int index
)

Set the key at the given index.

**Parameters:**

- `key` New key value
- `index` 0-based index of the array element to update

```cpp
void ArrayObject::setValue (string value, int index)
```

Set the value at the given index.

**Parameters:**

- `value` New array element value
- `index` 0-based index of the array element to update

```cpp
void ArrayObject::sort (bool ascending = false )
```

Alpha sorts the array by value.

**Parameters:**

- `ascending` [optional] True for ascending sort, false for descending sort

```cpp
void ArrayObject::sorta ( )
```

Alpha sorts the array by value in ascending order.

```cpp
void ArrayObject::sortd ( )
```
Alpha sorts the array by value in descending order.

```c
void ArrayObject::sortf(string functionName)
```

Sorts the array by value in ascending order using the given callback function.

**Parameters:**

- `functionName`: Name of a function that takes two arguments `A` and `B` and returns -1 if `A` is less, 1 if `B` is less, and 0 if both are equal.

**Example:**

```c
function mySortCallback(%a, %b)
{
    return strcmp( %a.name, %b.name );
}

%array.sortf( "mySortCallback" );
```

```c
void ArrayObject::sortfd(string functionName)
```

Sorts the array by value in descending order using the given callback function.

**Parameters:**

- `functionName`: Name of a function that takes two arguments `A` and `B` and returns -1 if `A` is less, 1 if `B` is less, and 0 if both are equal.

**See also:**

- `sortf`
### void ArrayObject::sortfk(string functionName)

Sorts the array by key in ascending order using the given callback function.

**Parameters:**

Name of a function that takes two arguments A and B and returns -1 if A is less, 1 if B is less, and 0 if both are equal.

**See also:**

sortf

### void ArrayObject::sortkd(string functionName)

Sorts the array by key in descending order using the given callback function.

**Parameters:**

Name of a function that takes two arguments A and B and returns -1 if A is less, 1 if B is less, and 0 if both are equal.

**See also:**

sortf

### void ArrayObject::sortk(bool ascending = false)

Alpha sorts the array by key.

**Parameters:**

`ascending` [optional] True for ascending sort, false for descending sort
void ArrayObject::sortka( )

Alpha sorts the array by key in ascending order.

void ArrayObject::sortkd( )

Alpha sorts the array by key in descending order.

void ArrayObject::sortn(bool ascending = false )

Numerically sorts the array by value.

**Parameters:**

\[
\text{ascending} \quad \text{[optional]} \quad \text{True for ascending sort, false for descending sort}
\]

void ArrayObject::sortna( )

Numerically sorts the array by value in ascending order.

void ArrayObject::sortnd( )

Numerically sorts the array by value in descending order.

void ArrayObject::sortnk(bool ascending = false )

Numerically sorts the array by key.

**Parameters:**
ascending [optional] True for ascending sort, false for descending sort

void ArrayObject::sortnka( )

Numerical sorts the array by key in ascending order.

void ArrayObject::sortnkd( )

Numerical sorts the array by key in descending order.

void ArrayObject::uniqueKey( )

Removes any elements that have duplicated keys (leaving the first instance).

void ArrayObject::uniqueValue( )

Removes any elements that have duplicated values (leaving the first instance).
Member Data Documentation

<table>
<thead>
<tr>
<th>bool ArrayObject::caseSensitive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Makes the keys and values case-sensitive.</td>
</tr>
<tr>
<td>By default, comparison of key and value strings will be case-insensitive.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>caseString ArrayObject::key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Helper field which allows you to add new key['keyname'] = value pairs.</td>
</tr>
</tbody>
</table>
BanList Class Reference

Used for kicking and banning players from a server. There is only a single instance of BanList. It is very important to note that you do not ever create this object in script like you would other game play objects. You simply reference it via namespace. More...

List of all members.
### Static Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>add (int uniqueld, string transportAddress, int banLength)</code></td>
<td>Ban a user for banLength seconds.</td>
</tr>
<tr>
<td><code>addAbsolute (int uniqueld, string transportAddress, int banTime)</code></td>
<td>Ban a user until a given time.</td>
</tr>
<tr>
<td><code>export (string filename)</code></td>
<td>Dump the banlist to a file.</td>
</tr>
<tr>
<td><code>isBanned (int uniqueld, string transportAddress)</code></td>
<td>Is someone banned?</td>
</tr>
<tr>
<td><code>removeBan (int uniqueld, string transportAddress)</code></td>
<td>Unban someone.</td>
</tr>
</tbody>
</table>
Detailed Description

Used for kicking and banning players from a server. There is only a single instance of BanList. It is very important to note that you do not ever create this object in script like you would other game play objects. You simply reference it via namespace.

For this to be used effectively, make sure you are hooking up other functions to BanList. For example, functions like GameConnection::onConnectRequestRejected( this, msg ) and function GameConnection::onConnectRequest are excellent places to make use of the BanList. Other systems can be used in conjunction for strict control over a server

See also:

   addBadWord
   containsBadWords
Member Function Documentation

static void BanList::add (int uniqueld, string transportAddress, int banLength) [static]

Ban a user for banLength seconds.

Parameters:

- uniqueld Unique ID of the player.
- transportAddress Address from which the player connected.
- banLength Time period over which to ban the player.

Example:

// Kick someone off the server
// %client - This is the connection to the
client

function kick (%client)
{
   // Let the server know what happened
   messageAll ( 'MsgAdminForce', '\c2The Admin has kicked %1.' );

   // If it is not an AI Player, execute the ban.
   if (!%client.isAIControlled())
      BanList::add (%client.guid, %client.getAddress(), $pref::Server::KickBanTime);

   // Let the player know they messed up.
   %client.delete ( "You have been kicked." );
}

static void BanList::addAbsolute (int uniqueld, string transportAddress,
Ban a user until a given time.

**Parameters:**

- **uniqueId** Unique ID of the player.
- **transportAddress** Address from which the player connected.
- **banTime** Time at which they will be allowed back in.

**Example:**

```cpp
// Kick someone off the server
// %client - This is the connection to the person we are kicking
function kick(%client)
{
    // Let the server know what happened
    messageAll('MsgAdminForce', '"The Admin has kicked %1."', %client.getName());

    // If it is not an AI Player, execute the ban.
    if (!%client.isAIControlled())
        BanList::addAbsolute(%client.guid, %client.getAddress(), $pref::Server::KickBanTime);

    // Let the player know they messed up
    %client.delete("You have been kicked.")
}
```

**static void BanList::export(string filename ) [static]**

Dump the banlist to a file.

**Parameters:**

- **filename** Path of the file to write the list to.
Example:

```cpp
BanList::Export("./server/banlist.cs");
```

```cpp
static bool BanList::isBanned(int uniqueld,
                               string transportAddress)
```

Is someone banned?

**Parameters:**

- `uniqueld`: Unique ID of the player.
- `transportAddress`: Address from which the player connected.

Example:

```cpp
//------------------------------------------------------------------------------
// This script function is called before a client connection is accepted. Returning will accept the
// connection. Anything else will be sent back as an error to the client. All the connect args will be passed
// also to the function GameConnection::onConnectRequest().
{
    // Find out who is trying to connect
    echo("Connect request from: " @ %netAddress);

    // Are they allowed in?
    if (BanList::isBanned(%client.guid, %netAddress))
        return "CR_YOUAREBANNED";

    // Is there room for an unbanned player?
    if ($Server::PlayerCount >= $pref::Server::MaxPlayers)
        return "CR_SERVERFULL";

    return ;
```
static void BanList::removeBan(int uniqueld, string transportAddress) [static]

Unban someone.

**Parameters:**
- *uniqueld*  
  Unique ID of the player.
- *transportAddress*  
  Address from which the player connected.

**Example:**

```cpp
BanList::removeBan(%userID, %ipAddress);
```

Copyright © GarageGames, LLC. All Rights Reserved.
BasicClouds Class Reference
[Atmosphere]

Renders up to three layers of scrolling cloud-cover textures overhead. More...

Inheritance diagram for BasicClouds:

```
SimObject
     ^
     |  NetObject
     |  SceneObject
     |  BasicClouds
```

List of all members.
Public Attributes

**BasicClouds**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>height</td>
<td>Abstract number which controls the curvature and height of the dome mesh.</td>
</tr>
<tr>
<td>bool</td>
<td>layerEnabled</td>
<td>Enable or disable rendering of this layer.</td>
</tr>
<tr>
<td>Point2F</td>
<td>texDirection</td>
<td>Texture scroll direction for this layer, relative to the world axis.</td>
</tr>
<tr>
<td>Point2F</td>
<td>texOffset</td>
<td>UV offset for this layer.</td>
</tr>
<tr>
<td>float</td>
<td>texScale</td>
<td>Texture repeat for this layer.</td>
</tr>
<tr>
<td>float</td>
<td>texSpeed</td>
<td>Texture scroll speed for this layer.</td>
</tr>
<tr>
<td>filename</td>
<td>texture</td>
<td>Texture for this layer.</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>static bool</th>
<th>isRenderable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static bool</th>
<th>isSelectable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Renders up to three layers of scrolling cloud-cover textures overhead.

BasicClouds always renders overhead, following the camera. It is intended as part of the background of your level, rendering in front of Sky/Sun type objects and behind everything else.

The parameters controlling the rendering of each texture are referred to and grouped as 'layers'. They are rendered in sequential order, so, layer 1 obscures layer 0, and so on.

**BasicClouds** is not affected by scene lighting and is therefore not appropriate for scenes in which lighting radically changes, such as day/night.
# Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>BasicClouds::height[3]</code> Abstract number which controls the curvature and height of the dome mesh.</td>
</tr>
<tr>
<td>bool</td>
<td><code>BasicClouds::layerEnabled[3]</code> Enable or disable rendering of this layer.</td>
</tr>
<tr>
<td>Point2F</td>
<td><code>BasicClouds::texDirection[3]</code> Texture scroll direction for this layer, relative to the world axis.</td>
</tr>
<tr>
<td>Point2F</td>
<td><code>BasicClouds::texOffset[3]</code> UV offset for this layer.</td>
</tr>
<tr>
<td>float</td>
<td><code>BasicClouds::texScale[3]</code> Texture repeat for this layer.</td>
</tr>
<tr>
<td>float</td>
<td><code>BasicClouds::texSpeed[3]</code> Texture scroll speed for this layer.</td>
</tr>
<tr>
<td>filename</td>
<td><code>BasicClouds::texture[3]</code></td>
</tr>
</tbody>
</table>
Texture for this layer.
Camera Class Reference
[Base Camera]

Represents a position, direction and field of view to render a scene from. More...

Inheritance diagram for Camera:

```
  SimObject
   /\     /
  /   \   /
 NetObject
      |
  "SceneObject"
      |
    "GameBase"
      |
    "ShapeBase"
      |
 Camera
```

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>autoFitRadius (float radius)</td>
<td>Move the camera to fully view the given radius.</td>
</tr>
<tr>
<td>VectorF</td>
<td>getAngularVelocity ()</td>
<td>Get the angular velocity for a Newton mode camera.</td>
</tr>
<tr>
<td>Camera::CameraMotionMode</td>
<td>getMode ()</td>
<td>Returns the current camera control mode.</td>
</tr>
<tr>
<td>Point3F</td>
<td>getOffset ()</td>
<td>Get the camera's offset from its orbit or tracking point.</td>
</tr>
<tr>
<td>Point3F</td>
<td>getPosition ()</td>
<td>Get the camera's position in the world.</td>
</tr>
<tr>
<td>Point3F</td>
<td>getRotation ()</td>
<td>Get the camera's Euler rotation in radians.</td>
</tr>
<tr>
<td>VectorF</td>
<td>getVelocity ()</td>
<td>Get the velocity for the camera.</td>
</tr>
<tr>
<td>bool</td>
<td>isEditOrbitMode ()</td>
<td>Is the camera in edit orbit mode?</td>
</tr>
<tr>
<td>bool</td>
<td>isRotationDamped ()</td>
<td>Is this a Newton Fly mode camera with damped rotation?</td>
</tr>
<tr>
<td>void</td>
<td>lookAt (Point3F point)</td>
<td>Point the camera at the specified position. Does not work in Orbit or Track modes.</td>
</tr>
<tr>
<td>void</td>
<td>setAngularDrag (float drag)</td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><code>void setAngularForce(float force)</code></td>
<td>Set the angular force for a Newton mode camera.</td>
<td></td>
</tr>
<tr>
<td><code>void setAngularVelocity(VectorF velocity)</code></td>
<td>Set the angular velocity for a Newton mode camera.</td>
<td></td>
</tr>
<tr>
<td><code>void setBrakeMultiplier(float multiplier)</code></td>
<td>Set the Newton mode camera brake multiplier when trigger[1] is active.</td>
<td></td>
</tr>
<tr>
<td><code>void setDrag(float drag)</code></td>
<td>Set the drag for a Newton mode camera.</td>
<td></td>
</tr>
<tr>
<td><code>void setEditOrbitMode()</code></td>
<td>Set the editor camera to orbit around a point set with <code>Camera::setEditOrbitPoint()</code>.</td>
<td></td>
</tr>
<tr>
<td><code>void setEditOrbitPoint(Point3F point)</code></td>
<td>Set the editor camera's orbit point.</td>
<td></td>
</tr>
<tr>
<td><code>void setFlyForce(float force)</code></td>
<td>Set the force applied to a Newton mode camera while moving.</td>
<td></td>
</tr>
<tr>
<td><code>void setFlyMode()</code></td>
<td>Set the camera to fly freely.</td>
<td></td>
</tr>
<tr>
<td><code>void setMass(float mass)</code></td>
<td>Set the mass for a Newton mode camera.</td>
<td></td>
</tr>
<tr>
<td><code>void setNewtonFlyMode()</code></td>
<td>Set the camera to fly freely, but with ease-in and ease-out.</td>
<td></td>
</tr>
<tr>
<td>Function Name</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>void setOffset (Point3F offset)</td>
<td>Set the camera's offset.</td>
<td></td>
</tr>
<tr>
<td>void setOrbitMode (GameBase orbitObject, TransformF orbitPoint, float minDistance, float maxDistance, float initDistance, bool ownClientObj=false, Point3F offset=Point3F(0.0f, 0.0f, 0.0f), bool locked=false)</td>
<td>Set the camera to orbit around the given object, or if none is given, around the given point.</td>
<td></td>
</tr>
<tr>
<td>bool setOrbitObject (GameBase orbitObject, VectorF rotation, float minDistance, float maxDistance, float initDistance, bool ownClientObject=false, Point3F offset=Point3F(0.0f, 0.0f, 0.0f), bool locked=false)</td>
<td>Set the camera to orbit around a given object.</td>
<td></td>
</tr>
<tr>
<td>void setOrbitPoint (TransformF orbitPoint, float minDistance, float maxDistance, float initDistance, Point3F offset=Point3F(0.0f, 0.0f, 0.0f), bool locked=false)</td>
<td>Set the camera to orbit around a given point.</td>
<td></td>
</tr>
<tr>
<td>void setRotation (Point3F rot)</td>
<td>Set the camera's Euler rotation in radians.</td>
<td></td>
</tr>
<tr>
<td>void setSpeedMultiplier (float multiplier)</td>
<td>Set the Newton mode camera speed multiplier when trigger[0] is active.</td>
<td></td>
</tr>
<tr>
<td>bool</td>
<td>setTrackObject (GameBase trackObject, Point3F offset=Point3F(0.0f, 0.0f, 0.0f))</td>
<td>Set the camera to track a given object.</td>
</tr>
<tr>
<td>------</td>
<td>-----------------------------------------------------------------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td>void</td>
<td>setValidEditOrbitPoint (bool validPoint)</td>
<td>Set if there is a valid editor camera orbit point.</td>
</tr>
<tr>
<td>void</td>
<td>setVelocity (VectorF velocity)</td>
<td>Set the velocity for the camera.</td>
</tr>
</tbody>
</table>
## Public Attributes

**Camera: Newton Mode**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>angularDrag</td>
<td>Drag on camera when rotating (Newton mode only). Default value is 2.</td>
</tr>
<tr>
<td>float</td>
<td>angularForce</td>
<td>Force applied on camera when asked to rotate (Newton mode only). Default value is 100.</td>
</tr>
<tr>
<td>float</td>
<td>brakeMultiplier</td>
<td>Speed multiplier when triggering the brake (Newton mode only). Default value is 2.</td>
</tr>
<tr>
<td>float</td>
<td>drag</td>
<td>Drag on camera when moving (Newton mode only). Default value is 2.</td>
</tr>
<tr>
<td>float</td>
<td>force</td>
<td>Force applied on camera when asked to move (Newton mode only). Default value is 500.</td>
</tr>
<tr>
<td>float</td>
<td>mass</td>
<td>The camera's mass (Newton mode only). Default value is 10.</td>
</tr>
<tr>
<td>bool</td>
<td>newtonMode</td>
<td>Apply smoothing (acceleration and damping) to camera movements.</td>
</tr>
<tr>
<td>bool</td>
<td>newtonRotation</td>
<td>Apply smoothing (acceleration and damping) to camera rotations.</td>
</tr>
<tr>
<td>float</td>
<td>speedMultiplier</td>
<td>Speed multiplier when triggering the speed.</td>
</tr>
</tbody>
</table>
accelerator (Newton mode only). Default value is 2.

Camera

<table>
<thead>
<tr>
<th>CameraMotionMode</th>
<th>controlMode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The current camera control mode.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td><strong>isRenderable</strong></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td><strong>isSelectable</strong></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>static float</td>
<td><strong>movementSpeed</strong></td>
<td>Global camera movement speed in units/s (typically m/s), with a base value of 40.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Represents a position, direction and field of view to render a scene from.

A camera is typically manipulated by a `GameConnection`. When set as the connection's control object, the camera handles all movement actions ($mvForwardAction, $mvPitch, etc.) just like a `Player`.

**Example:**

```cpp
// Set an already created camera as the GameConnection's control object.
connection.setControlObject(%camera);
```

**Methods of Operation**

The camera has two general methods of operation. The first is the standard mode where the camera starts and stops its motion and rotation instantly. This is the default operation of the camera and is used by most games. It may be specifically set with `Camera::setFlyMode()` for 6 DoF motion. It is also typically the method used with `Camera::setOrbitMode()` or one of its helper methods to orbit about a specific object (such as the Player's dead body) or a specific point.

The second method goes under the name of Newton as it follows Newton's 2nd law of motion: F=ma. This provides the camera with an ease-in and ease-out feel for both movement and rotation. To activate this method for movement, either use `Camera::setNewtonFlyMode()` or set the `Camera::newtonMode` field to true. To activate this method for rotation, set the `Camera::newtonRotation` to true. This method of operation is not typically used in games, and was developed to allow for a smooth fly through of a game level while recording a demo video. But with the right force and drag settings, it may give a more organic feel to the camera to games that use an overhead view, such as a RTS.

There is a third, minor method of operation but it is not generally used for games. This is when the camera is used with Torque's
World Editor in Edit Orbit Mode. When set, this allows the camera to rotate about a specific point in the world, and move towards and away from this point. See Camera::setEditOrbitMode() and Camera::setEditOrbitPoint(). While in this mode, Camera::autoFitRadius() may also be used.

Example:

```plaintext
// Create a camera in the level and set it
// Note: The camera starts in the standard
%cam = new Camera() {
    datablock = "Observer";
};
MissionCleanup.add( %cam );
%cam.setTransform( %spawnPoint.getTransform() );
```

Example:

```plaintext
// Create a camera at the given spawn point
// GameConnection i.e. the client. Uses the
// Sim::spawnObject() function to create the
// defined default settings.
// Note: The camera starts in the standard
function GameConnection::spawnCamera( %this ) {
    // Set the control object to the default
    if ( !isObject( %this.camera ) ) {
        if ( isDefined("$Game::DefaultCameraClass") ) {
            %this.camera = spawnObject( $Game::DefaultCameraClass,
        }
    // If we have a camera then set up some
    if ( isObject( %this.camera ) ) {
```
// Make sure we're cleaned up when the mission ends
MissionCleanup.add( %this.camera );

// Make sure the camera is always in scope
%this.camera.scopeToClient(%this);

// Send all user input from the connection to the camera
%this.setControlObject(%this.camera);

if (isDefined("%spawnPoint")) {
    // Attempt to treat %spawnPoint as an object, such as a SpawnSphere class.
    if (getWordCount(%spawnPoint) == 1) {
        %this.camera.setTransform(%spawnPoint.getTransform());
    }
    else {
        // Treat %spawnPoint as an AngleAxis transform
        %this.camera.setTransform(%spawnPoint);
    }
}

Motion Modes

Beyond the different operation methods, the Camera may be set to one of a number of motion modes. These motion modes determine how the camera will respond to input and may be used to constrain how the Camera moves. The CameraMotionMode enumeration defines the possible set of modes and the Camera's current may be obtained by using getMode().
Some of the motion modes may be set using specific script methods. These often provide additional parameters to set up the mode in one go. Otherwise, it is always possible to set a Camera’s motion mode using the controlMode property. Just pass in the name of the mode enum. The following table lists the motion modes, how to set them up, and what they offer:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Set From Script</th>
<th>Input Move</th>
<th>Input Rotate</th>
<th>Can Use Newton Mode?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stationary</td>
<td>controlMode property</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>FreeRotate</td>
<td>controlMode property</td>
<td>No</td>
<td>Yes</td>
<td>Rotate Only</td>
</tr>
<tr>
<td>Fly</td>
<td>setFlyMode()</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>OrbitObject</td>
<td>setOrbitMode()</td>
<td>Orbits object</td>
<td>Points to object</td>
<td>Move only</td>
</tr>
<tr>
<td>OrbitPoint</td>
<td>setOrbitPoint()</td>
<td>Orbits point</td>
<td>Points to location</td>
<td>Move only</td>
</tr>
<tr>
<td>TrackObject</td>
<td>setTrackObject()</td>
<td>No</td>
<td>Points to object</td>
<td>Yes</td>
</tr>
<tr>
<td>Overhead</td>
<td>controlMode property</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>EditOrbit (object selected)</td>
<td>setEditOrbitMode()</td>
<td>Orbits object</td>
<td>Points to object</td>
<td>Move only</td>
</tr>
<tr>
<td>EditOrbit (no object)</td>
<td>setEditOrbitMode()</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Trigger Input**

Passing a move trigger ($mvTriggerCount0, $mvTriggerCount1, etc.) on to a Camera performs different actions depending on which mode the camera is in. While in Fly, Overhead or EditOrbit mode, either trigger0 or trigger1 will cause a camera to move twice its normal movement speed. You can see this in action within the World Editor, where holding down the left mouse button while in mouse look mode (right mouse button is also down) causes the Camera to move faster.
Passing along trigger2 will put the camera into strafe mode. While in this mode a Fly, FreeRotate or Overhead Camera will not rotate from the move input. Instead the yaw motion will be applied to the Camera's x motion, and the pitch motion will be applied to the Camera's z motion. You can see this in action within the World Editor where holding down the middle mouse button allows the user to move the camera up, down and side-to-side.

While the camera is operating in Newton Mode, trigger0 and trigger1 behave slightly differently. Here trigger0 activates a multiplier to the applied acceleration force as defined by speedMultiplier. This has the affect of making the camera move up to speed faster. trigger1 has the opposite affect by acting as a brake. When trigger1 is active a multiplier is added to the Camera's drag as defined by brakeMultiplier.

See also:

CameraData
CameraMotionMode
Camera::movementSpeed
Member Function Documentation

void Camera::autoFitRadius(float radius)

Move the camera to fully view the given radius.

**Note:**

For this operation to take affect a valid edit orbit point must first be specified. See `Camera::setEditOrbitPoint()`.

**Parameters:**

`radius` The radius to view.

VectorF Camera::getAngularVelocity()

Get the angular velocity for a Newton mode camera.

**Returns:**

The angular velocity in the form of "x y z".

**Note:**

Only returns useful results when `Camera::newtonRotation` is set to true.

Camera::CameraMotionMode Camera::getMode()

Returns the current camera control mode.

**See also:**

`CameraMotionMode`

Point3F Camera::getOffset()
Get the camera's offset from its orbit or tracking point.

The offset is added to the camera's position when set to CameraMode::OrbitObject.

**Returns:**

The offset in the form of "x y z".

```cpp
Point3F Camera::getPosition()
```

Get the camera's position in the world.

**Returns:**

The position in the form of "x y z".

Reimplemented from SceneObject.

```cpp
Point3F Camera::getRotation()
```

Get the camera's Euler rotation in radians.

**Returns:**

The rotation in radians in the form of "x y z".

```cpp
VectorF Camera::getVelocity()
```

Get the velocity for the camera.

**Returns:**

The camera's velocity in the form of "x y z".

**Note:**

Only useful when the Camera is in Newton mode.
Reimplemented from ShapeBase.

**bool Camera::isEditOrbitMode()**

Is the camera in edit orbit mode?

**Returns:**
true if the camera is in edit orbit mode.

**bool Camera::isRotationDamped()**

Is this a Newton Fly mode camera with damped rotation?

**Returns:**
true if the camera uses a damped rotation. i.e. Camera::newtonRotation is set to true.

**void Camera::lookAt(Point3F point)**

Point the camera at the specified position. Does not work in Orbit or Track modes.

**Parameters:**
*point* The position to point the camera at.

**void Camera::setAngularDrag(float drag)**

Set the angular drag for a Newton mode camera.

**Parameters:**
*drag* The angular drag applied while the camera is rotating.
**Note:**
Only takes affect when `Camera::newtonRotation` is set to true.

```cpp
void Camera::setAngularForce(float force)
```

Set the angular force for a Newton mode camera.

**Parameters:**
- `force` The angular force applied when attempting to rotate the camera.

**Note:**
Only takes affect when `Camera::newtonRotation` is set to true.

```cpp
void Camera::setAngularVelocity(VectorF velocity)
```

Set the angular velocity for a Newton mode camera.

**Parameters:**
- `velocity` The angular velocity in form of "x y z".

**Note:**
Only takes affect when `Camera::newtonRotation` is set to true.

```cpp
void Camera::setBrakeMultiplier(float multiplier)
```

Set the Newton mode camera brake multiplier when trigger[1] is active.

**Parameters:**
- `multiplier` The brake multiplier to apply.

**Note:**
Only used when **Camera** is in Newton mode.

```cpp
void Camera::setDrag(float drag )
```

Set the drag for a Newton mode camera.

**Parameters:**

- `drag` The drag applied to the camera while moving.

**Note:**

Only used when **Camera** is in Newton mode.

```cpp
void Camera::setEditOrbitMode( )
```

Set the editor camera to orbit around a point set with `Camera::setEditOrbitPoint()`.

**Note:**

This method is generally used only within the World Editor and other tools. To orbit about an object or point within a game, see `Camera::setOrbitMode()` and its helper methods.

```cpp
void Camera::setEditOrbitPoint(Point3F point )
```

Set the editor camera's orbit point.

**Parameters:**

- `point` The point the camera will orbit in the form of "x y z".

```cpp
void Camera::setFlyForce(float force )
```

Set the force applied to a Newton mode camera while moving.
Parameters:

$force$ The force applied to the camera while attempting to move.

Note:

Only used when Camera is in Newton mode.

void Camera::setFlyMode(

Set the camera to fly freely.

Allows the camera to have 6 degrees of freedom. Provides for instantaneous motion and rotation unless one of the Newton fields has been set to true. See Camera::newtonMode and Camera::newtonRotation.

void Camera::setMass(float mass)

Set the mass for a Newton mode camera.

Parameters:

$mass$ The mass used during ease-in and ease-out calculations.

Note:

Only used when Camera is in Newton mode.

void Camera::setNewtonFlyMode(

Set the camera to fly freely, but with ease-in and ease-out.

This method allows for the same 6 degrees of freedom as Camera::setFlyMode() but activates the ease-in and ease-out on the camera's movement. To also activate Newton mode for the
camera's rotation, set `Camera::newtonRotation` to true.

```cpp
void Camera::setOffset(Point3F offset)
```

Set the camera's offset.

The offset is added to the camera's position when set to `CameraMode::OrbitObject`.

**Parameters:**

- `offset` The distance to offset the camera by in the form of "x y z".

```cpp
void Camera::setOrbitMode(GameBase orbitObject, TransformF orbitPoint, float minDistance, float maxDistance, float initDistance, bool ownClientObj = false, Point3F offset = Point3F(0.0f, 0.0f, 0.0f), bool locked = false)
```

Set the camera to orbit around the given object, or if none is given, around the given point.

**Parameters:**

- `orbitObject` The object to orbit around. If no object is given (0 or blank string is passed in) use the `orbitPoint` instead
- `orbitPoint` The point to orbit around when no object is given. In the form of "x y z ax ay az aa" such as returned by `SceneObject::getTransform()`.
- `minDistance` The minimum distance allowed to the orbit object or point.
**maxDistance**  
The maximum distance allowed from the orbit object or point.

**initDistance**  
The initial distance from the orbit object or point.

**ownClientObj**  
[optional] Are we orbiting an object that is owned by us? Default is false.

**offset**  
[optional] An offset added to the camera's position. Default is no offset.

**locked**  
[optional] Indicates the camera does not receive input from the player. Default is false.

See also:
- Camera::setOrbitObject()
- Camera::setOrbitPoint()

```cpp
bool Camera::setOrbitObject(GameBase orbitObject,  
    VectorF rotation,  
    float minDistance,  
    float maxDistance,  
    float initDistance,  
    bool ownClientObject = false,  
    Point3F offset = Point3F(0.0f, 0.0f, 0.0f),  
    bool locked = false)
```

Set the camera to orbit around a given object.

**Parameters:**

- **orbitObject**  
The object to orbit around.

- **rotation**  
The initial camera rotation about the object in radians in the form of "x y z".

- **minDistance**  
The minimum distance allowed to the orbit object or point.

- **maxDistance**  
The maximum distance allowed from the orbit object or point.
initDistance

The initial distance from the orbit object or point.

ownClientObject

[optional] Are we orbiting an object that is owned by us? Default is false.

offset

[optional] An offset added to the camera's position. Default is no offset.

locked

[optional] Indicates the camera does not receive input from the player. Default is false.

Returns:

false if the given object could not be found.

See also:

Camera::setOrbitMode()

```cpp
void Camera::setOrbitPoint(TransformF orbitPoint,
    float minDistance,
    float maxDistance,
    float initDistance,
    Point3F offset = Point3F(0.0f, 0.0f, 0.0f),
    bool locked = false
)
```

Set the camera to orbit around a given point.

Parameters:

**orbitPoint**

The point to orbit around. In the form of "x y z ax ay az aa" such as returned by SceneObject::getTransform().

**minDistance**

The minimum distance allowed to the orbit object or point.

**maxDistance**

The maximum distance allowed from the orbit object or point.

**initDistance**

The initial distance from the orbit object or point.
offset [optional] An offset added to the camera's position. Default is no offset.

locked [optional] Indicates the camera does not receive input from the player. Default is false.

See also:

Camera::setOrbitMode()

```cpp
void Camera::setRotation(Point3F rot)
```

Set the camera's Euler rotation in radians.

**Parameters:**

- `rot` The rotation in radians in the form of "x y z".

**Note:**

Rotation around the Y axis is ignored

```cpp
void Camera::setSpeedMultiplier(float multiplier)
```

Set the Newton mode camera speed multiplier when trigger[0] is active.

**Parameters:**

- `multiplier` The speed multiplier to apply.

**Note:**

Only used when `Camera` is in Newton mode.

```cpp
bool Camera::setTrackObject(GameBase trackObject, Point3F offset = Point3F(0.0f, 0.0f)
```
Set the camera to track a given object.

**Parameters:**

- `trackObject` The object to track.
- `offset` [optional] An offset added to the camera's position. Default is no offset.

**Returns:**

false if the given object could not be found.

```cpp
void Camera::setValidEditOrbitPoint (bool validPoint )
```

Set if there is a valid editor camera orbit point.

When `validPoint` is set to false the **Camera** operates as if it is in Fly mode rather than an Orbit mode.

**Parameters:**

- `validPoint` Indicates the validity of the orbit point.

**Note:**

Only used when **Camera** is in Edit Orbit Mode.

```cpp
void Camera::setVelocity(VectorF velocity )
```

Set the velocity for the camera.

**Parameters:**

- `velocity` The camera's velocity in the form of "x y z".

**Note:**

Only affects the **Camera** when in Newton mode.
Member Data Documentation

float Camera::angularDrag

Drag on camera when rotating (Newton mode only). Default value is 2.

float Camera::angularForce

Force applied on camera when asked to rotate (Newton mode only). Default value is 100.

float Camera::brakeMultiplier

Speed multiplier when triggering the brake (Newton mode only). Default value is 2.

CameraMotionMode Camera::controlMode

The current camera control mode.

float Camera::drag

Drag on camera when moving (Newton mode only). Default value is 2.

float Camera::force

Force applied on camera when asked to move (Newton mode only). Default value is 500.
**float Camera::mass**

The camera's mass (Newton mode only). Default value is 10.

**bool Camera::newtonMode**

Apply smoothing (acceleration and damping) to camera movements.

**bool Camera::newtonRotation**

Apply smoothing (acceleration and damping) to camera rotations.

**float Camera::speedMultiplier**

Speed multiplier when triggering the accelerator (Newton mode only). Default value is 2.

Copyright © GarageGames, LLC. All Rights Reserved.
CameraData Class Reference
[Base Camera]

A datablock that describes a camera. More...

Inheritance diagram for CameraData:

List of all members.
Detailed Description

A datablock that describes a camera.

Example:

datablock CameraData(Observer) {
    mode = "Observer";
};

See also:
Camera
Datablocks and Networking

Copyright © GarageGames, LLC. All Rights Reserved.
CloudLayer Class Reference
[Atmosphere]

A layer of clouds which change shape over time and are affected by scene lighting. More...

Inheritance diagram for CloudLayer:

List of all members.
### Public Attributes

#### CloudLayer

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorF</td>
<td>baseColor</td>
<td>Base cloud color before lighting.</td>
</tr>
<tr>
<td>float</td>
<td>coverage</td>
<td>Fraction of sky covered by clouds 0-1.</td>
</tr>
<tr>
<td>float</td>
<td>exposure</td>
<td>Brightness scale so CloudLayer can be overblown if desired.</td>
</tr>
<tr>
<td>float</td>
<td>height</td>
<td>Abstract number which controls the curvature and height of the dome mesh.</td>
</tr>
<tr>
<td>Point2F</td>
<td>texDirection</td>
<td>[3] Controls the direction this slot scrolls.</td>
</tr>
<tr>
<td>float</td>
<td>texScale</td>
<td>[3] Controls the texture repeat of this slot.</td>
</tr>
<tr>
<td>float</td>
<td>texSpeed</td>
<td>[3] Controls the speed this slot scrolls.</td>
</tr>
<tr>
<td>filename</td>
<td>texture</td>
<td>An RGBA texture which should contain normals and opacity (density).</td>
</tr>
<tr>
<td>float</td>
<td>windSpeed</td>
<td>Overall scalar to texture scroll speed.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>static bool</th>
<th>isRenderable</th>
<th>Disables rendering of all instances of this type.</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>


Detailed Description

A layer of clouds which change shape over time and are affected by scene lighting.

CloudLayer always renders overhead, following the camera. It is intended as part of the background of your level, rendering in front of Sky/Sun type objects and behind everything else.

The illusion of clouds forming and changing over time is controlled by the normal/opacity texture and the three sets of texture animation parameters. The texture is sampled three times. The first sample defines overall cloud density, where clouds are likely to form and their general size and shape. The second two samples control how it changes over time; they are combined and used as modifiers to the first sample.

CloudLayer is affected by scene lighting and is designed to be used in scenes with dynamic lighting or time of day changes.
## Member Data Documentation

<table>
<thead>
<tr>
<th><strong>ColorF</strong> CloudLayer::baseColor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base cloud color before lighting.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong> CloudLayer::coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fraction of sky covered by clouds 0-1.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong> CloudLayer::exposure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brightness scale so <strong>CloudLayer</strong> can be overblown if desired.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong> CloudLayer::height</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract number which controls the curvature and height of the dome mesh.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Point2F</strong> CloudLayer::texDirection[3]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controls the direction this slot scrolls.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong> CloudLayer::texScale[3]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controls the texture repeat of this slot.</td>
</tr>
</tbody>
</table>

| **float** CloudLayer::texSpeed[3] |
Controls the speed this slot scrolls.

**filename** CloudLayer::texture

An RGBA texture which should contain normals and opacity (density).

**float** CloudLayer::windSpeed

Overall scalar to texture scroll speed.

Copyright © GarageGames, LLC. All Rights Reserved.
ConsoleLogger Class Reference
[Logging]

Inheritance diagram for ConsoleLogger:

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>bool</th>
<th><strong>attach</strong> ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Attaches the logger to the console and begins writing to file.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th><strong>detach</strong> ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Detaches the logger from the console and stops writing to file.</td>
</tr>
</tbody>
</table>
Public Attributes

Logging

<table>
<thead>
<tr>
<th>LogLevel</th>
<th>level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

A class designed to be used as a console consumer and log the data it receives to a file.

See also:

dumpConsoleFunctions
dumpConsoleClasses
Member Function Documentation

bool ConsoleLogger::attach()

Attaches the logger to the console and begins writing to file.

Example:

```cpp
// Create the logger
// Will automatically start writing to testLogging.txt
new ConsoleLogger(logger, "testLogging.txt");

// Send something to the console, with the logger consuming and writing to file
echo("This is logged to the file");

// Stop logging, but do not delete the logger
logger.detach();

echo("This is not logged to the file");

// Attach the logger to the console again
logger.attach();

// Logging has resumed
echo("Logging has resumed");
```

bool ConsoleLogger::detach()

Detaches the logger from the console and stops writing to file.

Example:

```cpp
// Create the logger
// Will automatically start writing to testLogging.txt
```
new ConsoleLogger(logger, "testLogging.txt")

// Send something to the console, with the
echo("This is logged to the file");

// Stop logging, but do not delete the log
logger.detach();

echo("This is not logged to the file");

// Attach the logger to the console again
logger.attach();

// Logging has resumed
echo("Logging has resumed");
Member Data Documentation

LogLevel ConsoleLogger::level

Determines the priority level and attention the logged entry gets when recorded.

Copyright © GarageGames, LLC. All Rights Reserved.
ConvexShape Class Reference
[Miscellaneous]

A renderable, collidable convex shape defined by a collection of surface planes. More...

Inheritance diagram for ConvexShape:

```
SimObject
  ↓
NetObject
  ↓
SceneObject
  ↓
ConvexShape
```

[legend]

List of all members.
### Public Attributes

#### Rendering

<table>
<thead>
<tr>
<th>string</th>
<th>Material</th>
</tr>
</thead>
<tbody>
<tr>
<td>Material</td>
<td>Material used to render the ConvexShape surface.</td>
</tr>
</tbody>
</table>

#### Internal

<table>
<thead>
<tr>
<th>string</th>
<th>surface</th>
</tr>
</thead>
<tbody>
<tr>
<td>surface</td>
<td>Do not modify, for internal use.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
**Detailed Description**

A renderable, collidable convex shape defined by a collection of surface planes.

ConvexShape is intended to be used as a temporary asset for quickly blocking out a scene or filling in approximate shapes to be later replaced with final assets. This is most easily done by using the WorldEditor's Sketch Tool.
**Member Data Documentation**

**string** `ConvexShape::Material`

*Material* used to render the `ConvexShape` surface.

**string** `ConvexShape::surface`

Do not modify, for internal use.

Copyright © GarageGames, LLC. All Rights Reserved.
CubemapData Class Reference

[GFX]

Used to create static or dynamic cubemaps. More...

Inheritance diagram for CubemapData:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>string getFilename ()</code></td>
<td>Returns the script filename of where the CubemapData object was defined. This is used by the material editor.</td>
</tr>
<tr>
<td><code>void updateFaces ()</code></td>
<td>Update the assigned cubemaps faces.</td>
</tr>
</tbody>
</table>
### Public Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>dynamic</code> Set to true if this is a dynamic cubemap. The default is false.</td>
</tr>
<tr>
<td>float</td>
<td><code>dynamicFarDist</code> The far clip distance used when rendering to the dynamic cubemap.</td>
</tr>
<tr>
<td>float</td>
<td><code>dynamicNearDist</code> The near clip distance used when rendering to the dynamic cubemap.</td>
</tr>
<tr>
<td>int</td>
<td><code>dynamicObjectTypeMask</code> The typemask used to filter the objects rendered to the dynamic cubemap.</td>
</tr>
<tr>
<td>int</td>
<td><code>dynamicSize</code> The size of each dynamic cubemap face in pixels.</td>
</tr>
</tbody>
</table>
Detailed Description

Used to create static or dynamic cubemaps.

This object is used with Material, WaterObject, and other objects for cubemap reflections.

A simple declaration of a static cubemap:

Example:

```cpp
singleton CubemapData( SkyboxCubemap )
{
    cubeFace[0] = "./skybox_1";
    cubeFace[1] = "./skybox_2";
    cubeFace[2] = "./skybox_3";
    cubeFace[3] = "./skybox_4";
    cubeFace[4] = "./skybox_5";
    cubeFace[5] = "./skybox_6";
};
```

Note:

The dynamic cubemap functionality in CubemapData has been deprecated in favor of ReflectorDesc.

See also:

ReflectorDesc
Member Function Documentation

`string CubemapData::getFilename()`

Returns the script filename of where the `CubemapData` object was defined. This is used by the material editor.

Reimplemented from `SimObject`.

`void CubemapData::updateFaces()`

Update the assigned cubemaps faces.
### Member Data Documentation

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>filename</strong> CubemapData::cubeFace[6]</td>
<td>The 6 cubemap face textures for a static cubemap. They are in the following order:</td>
</tr>
<tr>
<td></td>
<td>- cubeFace[0] is -X</td>
</tr>
<tr>
<td></td>
<td>- cubeFace[1] is +X</td>
</tr>
<tr>
<td></td>
<td>- cubeFace[2] is -Z</td>
</tr>
<tr>
<td></td>
<td>- cubeFace[3] is +Z</td>
</tr>
<tr>
<td></td>
<td>- cubeFace[4] is -Y</td>
</tr>
<tr>
<td></td>
<td>- cubeFace[5] is +Y</td>
</tr>
<tr>
<td><strong>bool CubemapData::dynamic</strong></td>
<td>Set to true if this is a dynamic cubemap. The default is false.</td>
</tr>
<tr>
<td><strong>float CubemapData::dynamicFarDist</strong></td>
<td>The far clip distance used when rendering to the dynamic cubemap.</td>
</tr>
<tr>
<td><strong>float CubemapData::dynamicNearDist</strong></td>
<td>The near clip distance used when rendering to the dynamic cubemap.</td>
</tr>
<tr>
<td><strong>int CubemapData::dynamicObjectTypeMask</strong></td>
<td>The typemask used to filter the objects rendered to the dynamic</td>
</tr>
</tbody>
</table>
cubemap.

**int CubemapData::dynamicSize**

The size of each dynamic cubemap face in pixels.
CustomMaterial Class Reference

[Materials]

**Material** object which provides more control over surface properties.

More...

Inheritance diagram for CustomMaterial:

```
  SimObject
    ↓
  Material
    ↓
CustomMaterial
```

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Material</td>
<td>fallback</td>
<td>Alternate material for targeting lower end hardware.</td>
</tr>
<tr>
<td>bool</td>
<td>forwardLit</td>
<td>Determines if the material should recieve lights in Basic Lighting. Has no effect in Advanced Lighting.</td>
</tr>
<tr>
<td>string</td>
<td>shader</td>
<td>Name of the ShaderData to use for this effect.</td>
</tr>
<tr>
<td>GFXStateBlockData</td>
<td>stateBlock</td>
<td>Name of a GFXStateBlockData for this effect.</td>
</tr>
<tr>
<td>string</td>
<td>target</td>
<td>String identifier of this material's target texture.</td>
</tr>
<tr>
<td>float</td>
<td>version</td>
<td>Specifies pixel shader version for hardware.</td>
</tr>
</tbody>
</table>
**Detailed Description**

*Material* object which provides more control over surface properties.

CustomMaterials allow the user to specify their own shaders via the *ShaderData* datablock. Because CustomMaterials are derived from Materials, they can hold a lot of the same properties. It is up to the user to code how these properties are used.

**Example:**

```plaintext
singleton CustomMaterial( WaterBasicMat )
{
    sampler["reflectMap"] = "$reflectbuff";
    sampler["refractBuff"] = "$backbuff";

    cubemap = NewLevelSkyCubemap;
    shader = WaterBasicShader;
    stateBlock = WaterBasicStateBlock;
    version = 2.0;
}
```

See also: Material, GFXStateBlockData, ShaderData
Member Data Documentation

<table>
<thead>
<tr>
<th>Member</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Material CustomMaterial::fallback</td>
<td>Alternate material for targeting lower end hardware.</td>
</tr>
<tr>
<td></td>
<td>If the CustomMaterial requires a higher pixel shader version than the one it's using, it's fallback Material will be processed instead. If the fallback material wasn't defined, Torque 3D will assert and attempt to use a very basic material in it's place.</td>
</tr>
<tr>
<td>bool CustomMaterial::forwardLit</td>
<td>Determines if the material should recieve lights in Basic Lighting. Has no effect in Advanced Lighting.</td>
</tr>
<tr>
<td>string CustomMaterial::shader</td>
<td>Name of the ShaderData to use for this effect.</td>
</tr>
<tr>
<td>GFXStateBlockData CustomMaterial::stateBlock</td>
<td>Name of a GFXStateBlockData for this effect.</td>
</tr>
<tr>
<td>string CustomMaterial::target</td>
<td>String identifier of this material's target texture.</td>
</tr>
<tr>
<td>float CustomMaterial::version</td>
<td></td>
</tr>
</tbody>
</table>
Specifies pixel shader version for hardware.

Valid pixel shader versions include 2.0, 3.0, etc.

**Note:**
All features aren't compatible with all pixel shader versions.
Debris Class Reference
[Special Effects]

Base debris class. Uses the DebrisData datablock for properties of individual debris objects. More...

Inheritance diagram for Debris:

- SimObject
- NetObject
- SceneObject
- GameBase
- Debris

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Method</th>
<th>Signature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bool init (string inputPosition=&quot;1.0 1.0 1.0&quot;, string inputVelocity=&quot;1.0 0.0 0.0&quot;)</code></td>
<td>Manually set this piece of debris at the given position with the given velocity.</td>
<td></td>
</tr>
</tbody>
</table>
Public Attributes

Debris

float lifetime
   Length of time for this debris object to exist. When expired, the object will be deleted.
<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Base debris class. Uses the DebrisData datablock for properties of individual debris objects.

Debris is typically made up of a shape and up to two particle emitters. In most cases Debris objects are not created directly. They are usually produced automatically by other means, such as through the Explosion class. When an explosion goes off, its ExplosionData datablock determines what Debris to emit.

Example:

datablock ExplosionData(GrenadeLauncherExp) {
   // Assigning debris data
debris = GrenadeDebris;

   // Adjust how debris is ejected
debrisThetaMin = 10;
debrisThetaMax = 60;
debrisNum = 4;
debrisNumVariance = 2;
debrisVelocity = 25;
debrisVelocityVariance = 5;

   // Note: other ExplosionData properties
};

Note:

Debris are client side only objects.

See also:

DebrisData
ExplosionData
Explosion
Member Function Documentation

```cpp
bool Debris::init(string inputPosition = "1.0 1.0 1.0",
                  string inputVelocity = "1.0 0.0 0.0")
```

Manually set this piece of debris at the given position with the given velocity.

Usually you do not manually create Debris objects as they are generated through other means, such as an Explosion. This method exists when you do manually create a Debris object and want to have it start moving.

**Parameters:**

- `inputPosition` Position to place the debris.
- `inputVelocity` Velocity to move the debris after it has been placed.

**Returns:**

Always returns true.

**Example:**

```cpp
// Define the position
%position = "1.0 1.0 1.0";

// Define the velocity
%velocity = "1.0 0.0 0.0";

// Inform the debris object of its new position
%debris.init(%position,%velocity);
```
Member Data Documentation

**float Debris::lifetime**

Length of time for this debris object to exist. When expired, the object will be deleted.

The initial lifetime value comes from the DebrisData datablock.

See also:

DebrisData::lifetime
DebrisData::lifetimeVariance

Copyright © GarageGames, LLC. All Rights Reserved.
DebrisData Class Reference
[Special Effects]

Stores properties for an individual debris type. More...

Inheritance diagram for DebrisData:

List of all members.
## Public Attributes

### Physical Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>baseRadius</code></td>
<td>Radius at which the standard elasticity and friction apply.</td>
</tr>
<tr>
<td>int</td>
<td><code>bounceVariance</code></td>
<td>Allowed variance in the value of <code>numBounces</code>.</td>
</tr>
<tr>
<td>float</td>
<td><code>elasticity</code></td>
<td>A floating-point value specifying how 'bouncy' this object is.</td>
</tr>
<tr>
<td>float</td>
<td><code>friction</code></td>
<td>A floating-point value specifying how much velocity is lost to impact and sliding friction.</td>
</tr>
<tr>
<td>float</td>
<td><code>gravModifier</code></td>
<td>How much gravity affects debris.</td>
</tr>
<tr>
<td>float</td>
<td><code>lifetime</code></td>
<td>Amount of time until this debris object is destroyed.</td>
</tr>
<tr>
<td>float</td>
<td><code>lifetimeVariance</code></td>
<td>Allowed variance in the value of <code>lifetime</code>.</td>
</tr>
<tr>
<td>float</td>
<td><code>maxSpinSpeed</code></td>
<td>Maximum speed that this debris object will rotate.</td>
</tr>
<tr>
<td>float</td>
<td><code>minSpinSpeed</code></td>
<td>Minimum speed that this debris object will rotate.</td>
</tr>
<tr>
<td>int</td>
<td><code>numBounces</code></td>
<td>How many times to allow this debris object to bounce until it either explodes, becomes</td>
</tr>
</tbody>
</table>
static or snaps (defined in explodeOnMaxBounce, staticOnMaxBounce, snapOnMaxBounce).

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>float terminalVelocity</code></td>
<td>Max velocity magnitude.</td>
</tr>
<tr>
<td><code>bool useRadiusMass</code></td>
<td>Use mass calculations based on radius.</td>
</tr>
<tr>
<td><code>float velocity</code></td>
<td>Speed at which this debris object will move.</td>
</tr>
<tr>
<td><code>float velocityVariance</code></td>
<td>Allowed variance in the value of velocity.</td>
</tr>
</tbody>
</table>

**Datablocks**

<table>
<thead>
<tr>
<th>Datablock</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ParticleEmitterData</td>
<td><code>emitters [2]</code> List of particle emitters to spawn along with this debris object.</td>
</tr>
<tr>
<td>ExplosionData</td>
<td>`Explosion`` ExplosionData to spawn along with this debris object.</td>
</tr>
</tbody>
</table>

**Behavior**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bool explodeOnMaxBounce</code></td>
<td>If true, this debris object will explode after it has bounced max times.</td>
</tr>
<tr>
<td><code>bool fade</code></td>
<td>If true, this debris object will fade out when destroyed.</td>
</tr>
<tr>
<td><code>bool ignoreWater</code></td>
<td>If true, this debris object will not collide with water, acting as if the water is not there.</td>
</tr>
<tr>
<td><code>bool snapOnMaxBounce</code></td>
<td></td>
</tr>
</tbody>
</table>
If true, this debris object will snap into a resting position on the last bounce.

<table>
<thead>
<tr>
<th>bool</th>
<th>staticOnMaxBounce</th>
</tr>
</thead>
<tbody>
<tr>
<td>If true, this debris object becomes static after it has bounced max times.</td>
<td></td>
</tr>
</tbody>
</table>

**Display**

<table>
<thead>
<tr>
<th>filename</th>
<th>shapeFile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object model to use for this debris object.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>texture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Texture imagemap to use for this debris object.</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

Stores properties for an individual debris type.

DebrisData defines the base properties for a Debris object. Typically you'll want a Debris object to consist of a shape and possibly up to two particle emitters. The DebrisData datablock provides the definition for these items, along with physical properties and how a Debris object will react to other game objects, such as water and terrain.

Example:

datablock DebrisData(GrenadeDebris) {
    shapeFile = "art/shapes/weapons/ramrifle/debris.dts"
    emitters[0] = GrenadeDebrisFireEmitter;
    elasticity = 0.4;
    friction = 0.25;
    numBounces = 3;
    bounceVariance = 1;
    explodeOnMaxBounce = false;
    staticOnMaxBounce = false;
    snapOnMaxBounce = false;
    minSpinSpeed = 200;
    maxSpinSpeed = 600;
    lifetime = 4;
    lifetimeVariance = 1.5;
    velocity = 15;
    velocityVariance = 5;
    fade = true;
    useRadiusMass = true;
    baseRadius = 0.3;
    gravModifier = 1.0;
    terminalVelocity = 20;
}
ignoreWater = false;
}

See also:

Debris
**Member Data Documentation**

**float DebrisData::baseRadius**

Radius at which the standard elasticity and friction apply.

Only used when useRadiusMass is true.

**See also:**

useRadiusMass.

**int DebrisData::bounceVariance**

Allowed variance in the value of numBounces.

Must be less than numBounces.

**See also:**

numBounces

**float DebrisData::elasticity**

A floating-point value specifying how 'bouncy' this object is.

Must be in the range of -10 to 10.

**ParticleEmitterData DebrisData::emitters[2]**

List of particle emitters to spawn along with this debris object.

These are optional. You could have Debris made up of only a shape.
bool DebrisData::explodeOnMaxBounce

If true, this debris object will explode after it has bounced max times.

Be sure to provide an ExplosionData datablock for this to take effect.

See also:
  explosion

ExplosionData DebrisData::Explosion

ExplosionData to spawn along with this debris object.

This is optional as not all Debris explode.

bool DebrisData::fade

If true, this debris object will fade out when destroyed.

This fade occurs over the last second of the Debris' lifetime.

float DebrisData::friction

A floating-point value specifying how much velocity is lost to impact and sliding friction.

Must be in the range of -10 to 10.

float DebrisData::gravModifier

How much gravity affects debris.
### bool DebrisData::ignoreWater

If true, this debris object will not collide with water, acting as if the water is not there.

### float DebrisData::lifetime

Amount of time until this debris object is destroyed.

Must be in the range of 0 to 1000.

See also:
- lifetimeVariance

### float DebrisData::lifetimeVariance

Allowed variance in the value of lifetime.

Must be less than lifetime.

See also:
- lifetime

### float DebrisData::maxSpinSpeed

Maximum speed that this debris object will rotate.

Must be in the range of -10000 to 10000.

See also:
- minSpinSpeed
**float DebrisData::minSpinSpeed**

Minimum speed that this debris object will rotate.

Must be in the range of -10000 to 1000, and must be less than maxSpinSpeed.

**See also:**

maxSpinSpeed

**int DebrisData::numBounces**

How many times to allow this debris object to bounce until it either explodes, becomes static or snaps (defined in explodeOnMaxBounce, staticOnMaxBounce, snapOnMaxBounce).

Must be within the range of 0 to 10000.

**See also:**

bounceVariance

**filename DebrisData::shapeFile**

Object model to use for this debris object.

This shape is optional. You could have Debris made up of only particles.

**bool DebrisData::snapOnMaxBounce**

If true, this debris object will snap into a resting position on the last bounce.
bool DebrisData::staticOnMaxBounce

If true, this debris object becomes static after it has bounced max times.

float DebrisData::terminalVelocity

Max velocity magnitude.

string DebrisData::texture

Texture imagemap to use for this debris object.

Not used any more.

bool DebrisData::useRadiusMass

Use mass calculations based on radius.

Allows for the adjustment of elasticity and friction based on the Debris size.

See also:
   baseRadius

float DebrisData::velocity

Speed at which this debris object will move.

See also:
   velocityVariance
float DebrisData::velocityVariance

Allowed variance in the value of velocity.

Must be less than velocity.

See also:
  velocity

Copyright © GarageGames, LLC. All Rights Reserved.
DebugDrawer Class Reference

[GFX]

A debug helper for rendering debug primitives to the scene. More...

Inheritance diagram for DebugDrawer:

SimObject

DebugDrawer

[legend]

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void drawBox(Point3F a, Point3F b, ColorF color=ColorF::WHITE)</code></td>
<td>Draws an axis aligned box primitive within the two 3d points.</td>
</tr>
<tr>
<td><code>void drawLine(Point3F a, Point3F b, ColorF color=ColorF::WHITE)</code></td>
<td>Draws a line primitive between two 3d points.</td>
</tr>
<tr>
<td><code>void setLastTTL(int ms)</code></td>
<td>Sets the &quot;time to live&quot; (TTL) for the last rendered primitive.</td>
</tr>
<tr>
<td><code>void setLastZTest(bool enabled)</code></td>
<td>Sets the z buffer reading state for the last rendered primitive.</td>
</tr>
<tr>
<td><code>void toggleDrawing()</code></td>
<td>Toggles the rendering of <code>DebugDrawer</code> primitives.</td>
</tr>
<tr>
<td><code>void toggleFreeze()</code></td>
<td>Toggles freeze mode which keeps the currently rendered primitives from expiring.</td>
</tr>
</tbody>
</table>
**Detailed Description**

A debug helper for rendering debug primitives to the scene.

The **DebugDrawer** is used to render debug primitives to the scene for testing. It is often useful when debugging collision code or complex 3d algorithms to have them draw debug information, like culling hulls or bounding volumes, normals, simple lines, and so forth.

A key feature of the **DebugDrawer** is that each primitive gets a "time to live" (TTL) which allows them to continue to render to the scene for a fixed period of time. You can freeze or resume the system at any time to allow you to examine the output.

**Example:**

```plaintext
DebugDraw.drawLine( %player.getMuzzlePoint
DebugDraw.setLastTTL( 5000 ); // 5 seconds
```

The **DebugDrawer** renders solely in world space and all primitives are rendered with the cull mode disabled.

**Note:**

This feature can easily be used to cheat in online games, so you should be sure it is disabled in your shipping game. By default the **DebugDrawer** is disabled in all TORQUE_SHIPPING builds.
Member Function Documentation

```cpp
void DebugDrawer::drawBox(Point3F a,
                          Point3F b,
                          ColorF color = ColorF::WHITE)
```

Draws an axis aligned box primitive within the two 3d points.

```cpp
void DebugDrawer::drawLine(Point3F a,
                           Point3F b,
                           ColorF color = ColorF::WHITE)
```

Draws a line primitive between two 3d points.

```cpp
void DebugDrawer::setLastTTL(int ms)
```

Sets the "time to live" (TTL) for the last rendered primitive.

```cpp
void DebugDrawer::setLastZTest(bool enabled)
```

Sets the z buffer reading state for the last rendered primitive.

```cpp
void DebugDrawer::toggleDrawing()
```

Toggles the rendering of `DebugDrawer` primitives.

```cpp
void DebugDrawer::toggleFreeze()
```
Toggles freeze mode which keeps the currently rendered primitives from expiring.
DecalData Class Reference
[Decals, Special Effects]

A datablock describing an individual decal. More...

Inheritance diagram for DecalData:

SimObject

SimDataBlock

DecalData

[legend]

List of all members.
Public Member Functions

void postApply ()
    Recompute the imagemap sub-texture rectangles for this DecalData.
## Public Attributes

### Rendering

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>clippingAngle</td>
<td>The angle in degrees used to clip geometry that faces away from the decal projection direction.</td>
</tr>
<tr>
<td>float</td>
<td>fadeEndPixelSize</td>
<td>LOD value - size in pixels at which decals of this type are fully faded out.</td>
</tr>
<tr>
<td>float</td>
<td>fadeStartPixelSize</td>
<td>LOD value - size in pixels at which decals of this type begin to fade out.</td>
</tr>
<tr>
<td>char</td>
<td>renderPriority</td>
<td>Default renderPriority for decals of this type (determines draw order when decals overlap).</td>
</tr>
</tbody>
</table>

### Decal

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>fadeTime</td>
<td>Time (in milliseconds) over which to fade out the decal before deleting it at the end of its lifetime.</td>
</tr>
<tr>
<td>int</td>
<td>lifeSpan</td>
<td>Time (in milliseconds) before this decal will be automatically deleted.</td>
</tr>
<tr>
<td>string</td>
<td>Material</td>
<td>Material to use for this decal.</td>
</tr>
<tr>
<td>float</td>
<td>size</td>
<td>Width and height of the decal in meters before scale is applied.</td>
</tr>
</tbody>
</table>

### Texturing
<table>
<thead>
<tr>
<th><strong>int</strong> frame</th>
<th>Index of the texture rectangle within the imagemap to use for this decal.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>bool</strong> randomize</td>
<td>If true, a random frame from the imagemap is selected for each instance of the decal.</td>
</tr>
<tr>
<td><strong>int</strong> texCols</td>
<td>Number of columns in the supplied imagemap.</td>
</tr>
<tr>
<td><strong>int</strong> texRows</td>
<td>Number of rows in the supplied imagemap.</td>
</tr>
<tr>
<td><strong>int</strong> textureCoordCount</td>
<td>Number of individual frames in the imagemap (maximum 16).</td>
</tr>
<tr>
<td><strong>RectF</strong> textureCoords [16]</td>
<td>An array of RectFs (topleft.x topleft.y extent.x extent.y) representing the UV coordinates for each frame in the imagemap.</td>
</tr>
</tbody>
</table>
**Detailed Description**

A datablock describing an individual decal.

The textures defined by the decal **Material** can be divided into multiple rectangular sub-textures as shown below, with a different sub-texture selected by all decals using the same **DecalData** (via **frame**) or each decal instance (via **randomize**).

![Decal examples](image)

**Example of a Decal imagemap**

**Example:**

```plaintext
datablock DecalData(BulletHoleDecal)
{
    material = "DECAL_BulletHole";
    size = "5.0";
    lifeSpan = "50000";
    randomize = "1";
    texRows = "2";
    texCols = "2";
    clippingAngle = "60";
};
```

**See also:**
Decals
void DecalData::postApply()

Recompute the imagemap sub-texture rectangles for this DecalData.

Example:

// Inform the decal object to reload its:
%decalData.texRows = 4;
%decalData.postApply();
### Member Data Documentation

**float DecalData::clippingAngle**

The angle in degrees used to clip geometry that faces away from the decal projection direction.

**float DecalData::fadeEndPixelSize**

LOD value - size in pixels at which decals of this type are fully faded out.

This should be a smaller value than `fadeStartPixelSize`.

**float DecalData::fadeStartPixelSize**

LOD value - size in pixels at which decals of this type begin to fade out.

This should be a larger value than `fadeEndPixelSize`. However, you may also set this to a negative value to disable lod-based fading.

**int DecalData::fadeTime**

Time (in milliseconds) over which to fade out the decal before deleting it at the end of its lifetime.

**See also:**

- `lifeSpan`

**int DecalData::frame**
Index of the texture rectangle within the imagemap to use for this decal.

**int DecalData::lifeSpan**

Time (in milliseconds) before this decal will be automatically deleted.

**string DecalData::Material**

*Material* to use for this decal.

**bool DecalData::randomize**

If true, a random frame from the imagemap is selected for each instance of the decal.

**char DecalData::renderPriority**

Default renderPriority for decals of this type (determines draw order when decals overlap).

**float DecalData::size**

Width and height of the decal in meters before scale is applied.

**int DecalData::texCols**

Number of columns in the supplied imagemap.
Use `texRows` and `texCols` if the imagemap frames are arranged in a grid; use `textureCoords` to manually specify UV coordinates for irregular sized frames.

**int DecalData::texRows**

Number of rows in the supplied imagemap.

Use `texRows` and `texCols` if the imagemap frames are arranged in a grid; use `textureCoords` to manually specify UV coordinates for irregular sized frames.

**int DecalData::textureCoordCount**

Number of individual frames in the imagemap (maximum 16).

**RectF DecalData::textureCoords[16]**

An array of RectFs (topleft.x topleft.y extent.x extent.y) representing the UV coordinates for each frame in the imagemap.

**Note:**

This field should only be set if the imagemap frames are irregular in size. Otherwise use the `texRows` and `texCols` fields and the UV coordinates will be calculated automatically.
DecalManager Class Reference

[Decals, Special Effects]

The object that manages all of the decals in the active mission.

More...

Inheritance diagram for DecalManager:

```
  SimObject
    
  NetObject
    
  SceneObject
    
  DecalManager
```

[legend]

List of all members.
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td><strong>isRenderable</strong></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td><strong>isSelectable</strong></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

The object that manages all of the decals in the active mission.

See also:

Decals
DecalRoad Class Reference

[Terrain]

A strip shaped decal defined by spine nodes which clips against Terrain objects. More...

Inheritance diagram for DecalRoad:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>postApply()</code></td>
<td>Intended as a helper to developers and editor scripts.</td>
</tr>
<tr>
<td><code>regenerate()</code></td>
<td>Intended as a helper to developers and editor scripts.</td>
</tr>
</tbody>
</table>
# Public Attributes

## DecalRoad

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>breakAngle</td>
<td>Angle in degrees - DecalRoad will subdivide the spline if its curve is greater than this threshold.</td>
</tr>
<tr>
<td>string</td>
<td>Material</td>
<td>Material used for rendering.</td>
</tr>
<tr>
<td>int</td>
<td>renderPriority</td>
<td>DecalRoad(s) are rendered in descending renderPriority order.</td>
</tr>
<tr>
<td>float</td>
<td>textureLength</td>
<td>The length in meters of textures mapped to the DecalRoad.</td>
</tr>
</tbody>
</table>

## Internal

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>Node</td>
<td>Do not modify, for internal use.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>discardAll</td>
<td>For use by the Decal Editor.</td>
</tr>
<tr>
<td>static bool</td>
<td>EditorOpen</td>
<td>For use by the Decal Editor.</td>
</tr>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>showBatches</td>
<td>For use by the Decal Editor.</td>
</tr>
<tr>
<td>static bool</td>
<td>showRoad</td>
<td>For use by the Decal Editor.</td>
</tr>
<tr>
<td>static bool</td>
<td>showSpline</td>
<td>For use by the Decal Editor.</td>
</tr>
<tr>
<td>static int</td>
<td>updateDelay</td>
<td>For use by the Decal Editor.</td>
</tr>
<tr>
<td>static bool</td>
<td>wireframe</td>
<td>For use by the Decal Editor.</td>
</tr>
</tbody>
</table>
**Detailed Description**

A strip shaped decal defined by spine nodes which clips against Terrain objects.

*DecalRoad* is for representing a road or path (or other inventive things) across a *TerrainBlock*. It renders as a decal and is therefore only for features that do not need geometric depth.

The *Material* assigned to *DecalRoad* should tile vertically.
Member Function Documentation

```c
void DecalRoad::postApply()
```

Intended as a helper to developers and editor scripts.

Force trigger an inspectPostApply. This will transmit the material and other fields (not including nodes) to client objects.

```c
void DecalRoad::regenerate()
```

Intended as a helper to developers and editor scripts.

Force DecalRoad to update its spline and reclip geometry.
## Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>DecalRoad::breakAngle</code></td>
<td>Angle in degrees - <em>DecalRoad</em> will subdivide the spline if its curve is greater than this threshold.</td>
</tr>
<tr>
<td>bool</td>
<td><code>DecalRoad::discardAll</code></td>
<td>[static] For use by the Decal Editor.</td>
</tr>
<tr>
<td>bool</td>
<td><code>DecalRoad::EditorOpen</code></td>
<td>[static] For use by the Decal Editor.</td>
</tr>
<tr>
<td>string</td>
<td><code>DecalRoad::Material</code></td>
<td>Material used for rendering.</td>
</tr>
<tr>
<td>string</td>
<td><code>DecalRoad::Node</code></td>
<td>Do not modify, for internal use.</td>
</tr>
<tr>
<td>int</td>
<td><code>DecalRoad::renderPriority</code></td>
<td><em>DecalRoad</em>(s) are rendered in descending renderPriority order.</td>
</tr>
<tr>
<td>bool</td>
<td><code>DecalRoad::showBatches</code></td>
<td>[static]</td>
</tr>
</tbody>
</table>
bool DecalRoad::showRoad [static]

For use by the Decal Editor.

bool DecalRoad::showSpline [static]

For use by the Decal Editor.

float DecalRoad::textureLength

The length in meters of textures mapped to the DecalRoad.

int DecalRoad::updateDelay [static]

For use by the Decal Editor.

bool DecalRoad::wireframe [static]

For use by the Decal Editor.
EditorIconRegistry Class Reference

This class is used to find the correct icon file path for different SimObject class types. More...
**Detailed Description**

This class is used to find the correct icon file path for different SimObject class types.

It is typically used by the editors, not intended for actual game development.

Copyright © GarageGames, LLC. All Rights Reserved.
EventManager Class Reference

[Messaging]

The EventManager class is a wrapper for the standard messaging system. More...

Inheritance diagram for EventManager:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void dumpEvents ()</code></td>
<td>Print all registered events to the console.</td>
</tr>
<tr>
<td><code>void dumpSubscribers (String event)</code></td>
<td>Print all subscribers to an event to the console.</td>
</tr>
<tr>
<td><code>bool isRegisteredEvent (String event)</code></td>
<td>Check if an event is registered or not.</td>
</tr>
<tr>
<td><code>bool postEvent (String event, String data)</code></td>
<td>~Trigger an event.</td>
</tr>
<tr>
<td><code>bool registerEvent (String event)</code></td>
<td>Register an event with the event manager.</td>
</tr>
<tr>
<td><code>void remove (SimObject listener, String event)</code></td>
<td>Remove a listener from an event.</td>
</tr>
<tr>
<td><code>void removeAll (SimObject listener)</code></td>
<td>Remove a listener from all events.</td>
</tr>
<tr>
<td><code>bool subscribe (SimObject listener, String event, String callback)</code></td>
<td>Subscribe a listener to an event.</td>
</tr>
<tr>
<td><code>void unregisterEvent (String event)</code></td>
<td>Remove an event from the <code>EventManager</code>.</td>
</tr>
</tbody>
</table>
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>queue</td>
<td>List of events currently waiting.</td>
</tr>
</tbody>
</table>
Detailed Description

The EventManager class is a wrapper for the standard messaging system.

It provides functionality for management of event queues, events, and subscriptions. Creating an EventManager is as simple as calling new EventManager and specifying a queue name.

Example:

```php
// Create the EventManager.
$MyEventManager = new EventManager(); {
    queue
}

// Create an event.
$MyEventManager.registerEvent( "SomeCoolEvent" );

// Create a listener and subscribe.
$MyListener = new ScriptMsgListener(); {
    class
}$MyEventManager.subscribe( $MyListener, "SomeCoolEvent" );

function MyListener::onSomeCoolEvent( %this {
    echo( "onSomeCoolEvent Triggered" );
}

// Trigger the event.
$MyEventManager.postEvent( "SomeCoolEvent" );
```

See also:

ScriptMsgListener
void EventManager::dumpEvents( )

Print all registered events to the console.

void EventManager::dumpSubscribers(String event)

Print all subscribers to an event to the console.

**Parameters:**

*event*  
The event whose subscribers are to be printed. If this parameter isn't specified, all events will be dumped.

bool EventManager::isRegisteredEvent(String event)

Check if an event is registered or not.

**Parameters:**

*event*  
The event to check.

**Returns:**

Whether or not the event exists.

bool EventManager::postEvent(String event, String data)

~Trigger an event.

**Parameters:**

*event*  
The event to trigger.
data  The data associated with the event.

**Returns:**
Whether or not the event was dispatched successfully.

```cpp
bool EventManager::registerEvent(String event)
```

Register an event with the event manager.

**Parameters:**
- `event`  The event to register.

**Returns:**
Whether or not the event was registered successfully.

```cpp
void EventManager::remove(SimObject listener, String event)
```

Remove a listener from an event.

**Parameters:**
- `listener`  The listener to remove.
- `event`  The event to be removed from.

```cpp
void EventManager::removeAll(SimObject listener)
```

Remove a listener from all events.

**Parameters:**
- `listener`  The listener to remove.
bool EventManager::subscribe(SimObject listener, String event, String callback)

Subscribe a listener to an event.

**Parameters:**

- **listener**  The listener to subscribe.
- **event**    The event to subscribe to.
- **callback** Optional method name to receive the event notification. If this is not specified, "on[event]" will be used.

**Returns:**

Whether or not the subscription was successful.

void EventManager::unregisterEvent(String event)

Remove an event from the EventManager.

**Parameters:**

- **event**  The event to remove.
Member Data Documentation

string EventManager::queue

List of events currently waiting.

Copyright © GarageGames, LLC. All Rights Reserved.
Explosion Class Reference
[Special Effects]

The emitter for an explosion effect, with properties defined by a ExplosionData object. More...

Inheritance diagram for Explosion:

```
Legend:
- ▲: Inheritance

SimObject
      ▲
       ▲
    NetObject
          ▲
           ▲
        SceneObject
              ▲
               ▲
            GameBase
                  ▲
                   ▲
                 Explosion
```

List of all members.
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

The emitter for an explosion effect, with properties defined by a ExplosionData object.

The object will initiate the explosion effects automatically after being added to the simulation.

Example:

datablock ExplosionData( GrenadeSubExplosion )
{
    offset = 0.25;
    emitter[0] = GrenadeExpSparkEmitter;

    lightStartRadius = 4.0;
    lightEndRadius = 0.0;
    lightStartColor = "0.9 0.7 0.7";
    lightEndColor = "0.9 0.7 0.7";
    lightStartBrightness = 2.0;
    lightEndBrightness = 0.0;
};

datablock ExplosionData( GrenadeLauncherExplosion )
{
    soundProfile = GrenadeLauncherExplosionSound;
    lifeTimeMS = 400; // Quick flash, short

    // Volume particles
    particleEmitter = GrenadeExpFireEmitter
    particleDensity = 75;
    particleRadius = 2.25;

    // Point emission
    emitter[0] = GrenadeExpDustEmitter;
emitter[1] = GrenadeExpSparksEmitter;

// Sub explosion objects
subExplosion[0] = GrenadeSubExplosion;

// Camera Shaking
shakeCamera = true;
camShakeFreq = "10.0 11.0 9.0";
camShakeAmp = "15.0 15.0 15.0";
camShakeDuration = 1.5;
camShakeRadius = 20;

// Exploding debris
debris = GrenadeDebris;
debrisThetaMin = 10;
debrisThetaMax = 60;
debrisNum = 4;
debrisNumVariance = 2;
debrisVelocity = 25;
debrisVelocityVariance = 5;

lightStartRadius = 4.0;
lightEndRadius = 0.0;
lightStartColor = "1.0 1.0 1.0";
lightEndColor = "1.0 1.0 1.0";
lightStartBrightness = 4.0;
lightEndBrightness = 0.0;
lightNormalOffset = 2.0;

function createExplosion()
{
}
// Create a new explosion - it will exp.
%pos = "0 0 100";
%obj = new Explosion()
{
    position = %pos;
    dataBlock = GrenadeLauncherExplosion
};

// schedule an explosion
schedule(1000, 0, createExplosion);
ExplosionData Class Reference
[Special Effects]

Defines the attributes of an Explosion: particleEmitters, debris, lighting and camera shake effects. More...

Inheritance diagram for ExplosionData:

```
+-------------------+    +-------------------+
| SimObject         |    | SimDataBlock       |
|                   |    | GameBaseData       |
|                   |    | ExplosionData      |
```

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point3F</td>
<td>camShakeAmp</td>
<td>Amplitude of camera shaking, defined in the &quot;X Y Z&quot; axes.</td>
</tr>
<tr>
<td>float</td>
<td>camShakeDuration</td>
<td>Duration (in seconds) to shake the camera.</td>
</tr>
<tr>
<td>float</td>
<td>camShakeFalloff</td>
<td>Falloff value for the camera shake.</td>
</tr>
<tr>
<td>Point3F</td>
<td>camShakeFreq</td>
<td>Frequency of camera shaking, defined in the &quot;X Y Z&quot; axes.</td>
</tr>
<tr>
<td>float</td>
<td>camShakeRadius</td>
<td>Radial distance that a camera's position must be within relative to the center of the explosion to be shaken.</td>
</tr>
<tr>
<td>DebrisData</td>
<td>Debris</td>
<td>List of DebrisData objects to spawn with this explosion.</td>
</tr>
<tr>
<td>int</td>
<td>debrisNum</td>
<td>Number of debris objects to create.</td>
</tr>
<tr>
<td>int</td>
<td>debrisNumVariance</td>
<td>Variance in the number of debris objects to create (must be from 0 - debrisNum).</td>
</tr>
<tr>
<td>float</td>
<td>debrisPhiMax</td>
<td>Maximum reference angle, from the vertical plane, to eject debris from.</td>
</tr>
<tr>
<td>float</td>
<td>debrisPhiMin</td>
<td>Minimum reference angle, from the vertical plane, to eject debris from.</td>
</tr>
<tr>
<td>float</td>
<td>debrisThetaMax</td>
<td>Maximum angle, from the horizontal plane, to</td>
</tr>
</tbody>
</table>
eject debris from.

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>debrisThetaMin</td>
<td>Minimum angle, from the horizontal plane, to eject debris from.</td>
</tr>
<tr>
<td>float</td>
<td>debrisVelocity</td>
<td>Velocity to toss debris at.</td>
</tr>
<tr>
<td>float</td>
<td>debrisVelocityVariance</td>
<td>Variance in the debris initial velocity (must be ( \geq 0 )).</td>
</tr>
<tr>
<td>int</td>
<td>delayMS</td>
<td>Amount of time, in milliseconds, to delay the start of the explosion effect from the creation of the Explosion object.</td>
</tr>
<tr>
<td>int</td>
<td>delayVariance</td>
<td>Variance, in milliseconds, of delayMS.</td>
</tr>
<tr>
<td>ParticleEmitterData</td>
<td>emitter [4]</td>
<td>List of additional ParticleEmitterData objects to spawn with this explosion.</td>
</tr>
<tr>
<td>Point3F</td>
<td>explosionScale</td>
<td>&quot;X Y Z&quot; scale factor applied to the explosionShape model at the start of the explosion.</td>
</tr>
<tr>
<td>filename</td>
<td>explosionShape</td>
<td>Optional DTS or DAE shape to place at the center of the explosion.</td>
</tr>
<tr>
<td>bool</td>
<td>faceViewer</td>
<td>Controls whether the visual effects of the explosion always face the camera.</td>
</tr>
<tr>
<td>int</td>
<td>lifetimeMS</td>
<td>Lifetime, in milliseconds, of the Explosion object.</td>
</tr>
<tr>
<td>int</td>
<td>lifetimeVariance</td>
<td></td>
</tr>
</tbody>
</table>
Variance, in milliseconds, of the lifetimeMS of the *Explosion* object.

**float** `lightEndBrightness`  
Final brightness of the *PointLight* created by this explosion.

**ColorF** `lightEndColor`  
Final color of the *PointLight* created by this explosion.

**float** `lightEndRadius`  
Final radius of the *PointLight* created by this explosion.

**float** `lightNormalOffset`  
Distance (in the explosion normal direction) of the *PointLight* position from the explosion center.

**float** `lightStartBrightness`  
Initial brightness of the *PointLight* created by this explosion.

**ColorF** `lightStartColor`  
Initial color of the *PointLight* created by this explosion.

**float** `lightStartRadius`  
Initial radius of the *PointLight* created by this explosion.

**float** `offset`  
Offset distance (in a random direction) of the center of the explosion from the *Explosion* object position.

**int** `particleDensity`  
Density of the particle cloud created at the start of the explosion.

**ParticleEmitterData** `ParticleEmitter`  
Emitter used to generate a cloud of particles
at the start of the explosion.

<table>
<thead>
<tr>
<th>Type</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>particleRadius</code></td>
<td>Radial distance from the explosion center at which cloud particles are emitted.</td>
</tr>
<tr>
<td>float</td>
<td><code>playSpeed</code></td>
<td>Time scale at which to play the explosionShape ambient sequence.</td>
</tr>
<tr>
<td>bool</td>
<td><code>shakeCamera</code></td>
<td>Controls whether the camera shakes during this explosion.</td>
</tr>
<tr>
<td>Point3F</td>
<td><code>sizes [4]</code></td>
<td>&quot;X Y Z&quot; size keyframes used to scale the explosionShape model.</td>
</tr>
<tr>
<td>SFXTrack</td>
<td><code>soundProfile</code></td>
<td>Non-looping sound effect that will be played at the start of the explosion.</td>
</tr>
<tr>
<td>ExplosionData</td>
<td><code>subExplosion [5]</code></td>
<td>List of additional ExplosionData objects to create at the start of the explosion.</td>
</tr>
<tr>
<td>float</td>
<td><code>times [4]</code></td>
<td>Time keyframes used to scale the explosionShape model.</td>
</tr>
</tbody>
</table>
Detailed Description

Defines the attributes of an Explosion: particleEmitters, debris, lighting and camera shake effects.
Member Data Documentation

Point3F ExplosionData::camShakeAmp

Amplitude of camera shaking, defined in the "X Y Z" axes. Set any value to 0 to disable shaking in that axis.

float ExplosionData::camShakeDuration

Duration (in seconds) to shake the camera.

float ExplosionData::camShakeFalloff

Falloff value for the camera shake.

Point3F ExplosionData::camShakeFreq

Frequency of camera shaking, defined in the "X Y Z" axes.

float ExplosionData::camShakeRadius

Radial distance that a camera's position must be within relative to the center of the explosion to be shaken.

DebrisData ExplosionData::Debris

List of DebrisData objects to spawn with this explosion.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int ExplosionData::debrisNum</td>
<td>Number of debris objects to create.</td>
</tr>
<tr>
<td>int ExplosionData::debrisNumVariance</td>
<td>Variance in the number of debris objects to create (must be from 0 - debrisNum).</td>
</tr>
<tr>
<td>float ExplosionData::debrisPhiMax</td>
<td>Maximum reference angle, from the vertical plane, to eject debris from.</td>
</tr>
<tr>
<td>float ExplosionData::debrisPhiMin</td>
<td>Minimum reference angle, from the vertical plane, to eject debris from.</td>
</tr>
<tr>
<td>float ExplosionData::debrisThetaMax</td>
<td>Maximum angle, from the horizontal plane, to eject debris from.</td>
</tr>
<tr>
<td>float ExplosionData::debrisThetaMin</td>
<td>Minimum angle, from the horizontal plane, to eject debris from.</td>
</tr>
<tr>
<td>float ExplosionData::debrisVelocity</td>
<td>Velocity to toss debris at.</td>
</tr>
</tbody>
</table>
**float ExplosionData::debrisVelocityVariance**

Variance in the debris initial velocity (must be >= 0).

**int ExplosionData::delayMS**

Amount of time, in milliseconds, to delay the start of the explosion effect from the creation of the Explosion object.

**int ExplosionData::delayVariance**

Variance, in milliseconds, of delayMS.

**ParticleEmitterData ExplosionData::emitter[4]**

List of additional ParticleEmitterData objects to spawn with this explosion.

**See also:**

particleEmitter

**Point3F ExplosionData::explosionScale**

"X Y Z" scale factor applied to the explosionShape model at the start of the explosion.

**filename ExplosionData::explosionShape**

Optional DTS or DAE shape to place at the center of the
explosion.

The *ambient* animation of this model will be played automatically at the start of the explosion.

<table>
<thead>
<tr>
<th>bool ExplosionData::faceViewer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controls whether the visual effects of the explosion always face the camera.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int ExplosionData::lifetimeMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lifetime, in milliseconds, of the Explosion object.</td>
</tr>
</tbody>
</table>

**Note:**
If explosionShape is defined and contains an *ambient* animation, this field is ignored, and the playSpeed scaled duration of the animation is used instead.

<table>
<thead>
<tr>
<th>int ExplosionData::lifetimeVariance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variance, in milliseconds, of the lifetimeMS of the Explosion object.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float ExplosionData::lightEndBrightness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Final brightness of the PointLight created by this explosion.</td>
</tr>
</tbody>
</table>

**See also:**
lightStartBrightness

| ColorF ExplosionData::lightEndColor |
Final color of the PointLight created by this explosion.

**See also:**
- lightStartColor

---

```plaintext
float ExplosionData::lightEndRadius
```

Final radius of the PointLight created by this explosion.

**See also:**
- lightStartRadius

---

```plaintext
float ExplosionData::lightNormalOffset
```

Distance (in the explosion normal direction) of the PointLight position from the explosion center.

---

```plaintext
float ExplosionData::lightStartBrightness
```

Initial brightness of the PointLight created by this explosion.

Brightness is linearly interpolated from lightStartBrightness to lightEndBrightness over the lifetime of the explosion.

**See also:**
- lifetimeMS

---

```plaintext
ColorF ExplosionData::lightStartColor
```

Initial color of the PointLight created by this explosion.
Color is linearly interpolated from `lightStartColor` to `lightEndColor` over the lifetime of the explosion.

**See also:**
- `lifetimeMS`

```plaintext
float ExplosionData::lightStartRadius
```

Initial radius of the `PointLight` created by this explosion.

Radius is linearly interpolated from `lightStartRadius` to `lightEndRadius` over the lifetime of the explosion.

**See also:**
- `lifetimeMS`

```plaintext
float ExplosionData::offset
```

Offset distance (in a random direction) of the center of the explosion from the `Explosion` object position.

Most often used to create some variance in position for subExplosion effects.

```plaintext
int ExplosionData::particleDensity
```

Density of the particle cloud created at the start of the explosion.

**See also:**
- `particleEmitter`

```plaintext
ParticleEmitterData ExplosionData::ParticleEmitter
```
Emitter used to generate a cloud of particles at the start of the explosion.

Explosions can generate two different particle effects. The first is a single burst of particles at the start of the explosion emitted in a spherical cloud using particleEmitter.

The second effect spawns the list of ParticleEmitters given by the emitter[] field. These emitters generate particles in the normal way throughout the lifetime of the explosion.

```plaintext
float ExplosionData::particleRadius
```

Radial distance from the explosion center at which cloud particles are emitted.

**See also:**

particleEmitter

```plaintext
float ExplosionData::playSpeed
```

Time scale at which to play the explosionShape ambient sequence.

```plaintext
bool ExplosionData::shakeCamera
```

Controls whether the camera shakes during this explosion.

```plaintext
Point3F ExplosionData::sizes[4]
```

"X Y Z" size keyframes used to scale the explosionShape model.
The explosionShape (if defined) will be scaled using the times/sizes keyframes over the lifetime of the explosion.

See also:

 lifetimeMS

**SFXTrack ExplosionData::soundProfile**

Non-looping sound effect that will be played at the start of the explosion.

**ExplosionData ExplosionData::subExplosion[5]**

List of additional ExplosionData objects to create at the start of the explosion.

**float ExplosionData::times[4]**

Time keyframes used to scale the explosionShape model.

Values should be in increasing order from 0.0 - 1.0, and correspond to the life of the Explosion where 0 is the beginning and 1 is the end of the explosion lifetime.

See also:

 lifetimeMS

Copyright © GarageGames, LLC. All Rights Reserved.
FileDialog Class Reference
[File I/O]

Base class responsible for displaying an OS file browser. More...

Inheritance diagram for FileDialog:

SimObject

FileDialog

OpenFileDialog  SaveFileDialog

OpenFolderDialog

List of all members.
Public Member Functions

bool Execute ()
Launches the OS file browser.
# Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>changePath</code></td>
<td>True/False whether to set the working directory to the directory returned by the dialog.</td>
</tr>
<tr>
<td>string</td>
<td><code>defaultFile</code></td>
<td>The default file path when the dialog is shown.</td>
</tr>
<tr>
<td>string</td>
<td><code>defaultPath</code></td>
<td>The default directory path when the dialog is shown.</td>
</tr>
<tr>
<td>string</td>
<td><code>fileName</code></td>
<td>The default file name when the dialog is shown.</td>
</tr>
<tr>
<td>string</td>
<td><code>filters</code></td>
<td>The filter string for limiting the types of files visible in the dialog. It makes use of the pipe symbol '</td>
</tr>
<tr>
<td>string</td>
<td><code>title</code></td>
<td>The title for the dialog.</td>
</tr>
</tbody>
</table>
Detailed Description

Base class responsible for displaying an OS file browser.

FileDialog is a platform agnostic dialog interface for querying the user for file locations. It is designed to be used through the exposed scripting interface.

FileDialog is the base class for Native File Dialog controls in Torque. It provides these basic areas of functionality:

- Inherits from SimObject and is exposed to the scripting interface
- Provides blocking interface to allow instant return to script execution
- Simple object configuration makes practical use easy and effective

FileDialog is *NOT* intended to be used directly in script and is only exposed to script to expose generic file dialog attributes.

This base class is usable in TorqueScript, but does not specify what functionality is intended (open or save?). Its children, OpenFileDialog and SaveFileDialog, do make use of DialogStyle flags and do make use of specific functionality. These are the preferred classes to use.

However, the FileDialog base class does contain the key properties and important method for file browsing. The most important function is Execute(). This is used by both SaveFileDialog and OpenFileDialog to initiate the browser.

Example:

```javascript
// NOTE: This is not the preferred class to use

// Create the file dialog
%baseFileDialog = new FileDialog()
{
  // Allow browsing of all file types
```
filters = "*.*";

// No default file
defaultFile = ;

// Set default path relative to project
defaultPath = "./";

// Set the title
title = "Durpa";

// Allow changing of path you are browsing
changePath = true;
}

// Launch the file dialog
%baseFileDialog.Execute();

// Don't forget to cleanup
%baseFileDialog.delete();

Note:
FileDialog and its related classes are only available in a Tools build of Torque.

See also:
OpenFileDialog for a practical example on opening a file
SaveFileDialog for a practical example of saving a file
bool FileDialog::Execute()

Launches the OS file browser.

After an Execute() call, the chosen file name and path is available in one of two areas. If only a single file selection is permitted, the results will be stored in the fileName attribute.

If multiple file selection is permitted, the results will be stored in the files array. The total number of files in the array will be stored in the fileCount attribute.

Example:

```cpp
// NOTE: This is not the preferred class to use

// Create the file dialog
%baseFileDialog = new FileDialog()
{
    // Allow browsing of all file types
    filters = "*.*";

    // No default file
    defaultFile = ;

    // Set default path relative to project
    defaultPath = "./";

    // Set the title
    title = "Durpa";

    // Allow changing of path you are browsing
    changePath = true;
```
// Launch the file dialog
%baseFileDialog.Execute();

// Don't forget to cleanup
%baseFileDialog.delete();

// A better alternative is to use the derived classes which are specific to

// Create a dialog dedicated to opening images
%openFileDlg = new OpenFileDialog()
{
    // Look for jpg image files
    // First part is the descriptor|second part is the extension
    Filters = "Jepg Files|*.jpg";
    // Allow browsing through other folders
    ChangePath = true;

    // Only allow opening of one file at a time
    MultipleFiles = false;
};

// Launch the open file dialog
%result = %openFileDlg.Execute();

// Obtain the chosen file name and path
if ( %result )
{
    %selectedFile = %openFileDlg.file;
}
else
{
    %selectedFile = "";
}
// Cleanup
%openFileDialog.delete();

// Create a dialog dedicated to saving a file
%saveFileDialog = new SaveFileDialog()
{
    // Only allow for saving of COLLADA files
    Filters = "COLLADA Files (*.dae)|*.dae"

    // Default save path to where the WorldEditor last saved
    DefaultPath = $pref::WorldEditor::LastPath;

    // No default file specified
    DefaultFile = "";

    // Do not allow the user to change to a new directory
    ChangePath = false;

    // Prompt the user if they are going to overwrite an existing file
    OverwritePrompt = true;
}

// Launch the save file dialog
%result = %saveFileDialog.Execute();

// Obtain the file name
%selectedFile = "";
if ( %result )
%selectedFile = %saveFileDlg.file;

// Cleanup
%saveFileDlg.delete();

Returns:
True if the file was selected was successfully found (opened) or declared (saved).
### Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><strong>FileDialog::changePath</strong>&lt;br&gt;True/False whether to set the working directory to the directory returned by the dialog.</td>
</tr>
<tr>
<td>string</td>
<td><strong>FileDialog::defaultFile</strong>&lt;br&gt;The default file path when the dialog is shown.</td>
</tr>
<tr>
<td>string</td>
<td><strong>FileDialog::defaultPath</strong>&lt;br&gt;The default directory path when the dialog is shown.</td>
</tr>
<tr>
<td>string</td>
<td><strong>FileDialog::fileName</strong>&lt;br&gt;The default file name when the dialog is shown.</td>
</tr>
<tr>
<td>string</td>
<td><strong>FileDialog::filters</strong>&lt;br&gt;The filter string for limiting the types of files visible in the dialog. It makes use of the pipe symbol ‘</td>
</tr>
<tr>
<td>string</td>
<td><strong>FileDialog::title</strong></td>
</tr>
</tbody>
</table>
The title for the dialog.
**FileObject Class Reference**

[File I/O]

This class is responsible opening, reading, creating, and saving file contents. More...

Inheritance diagram for FileObject:

```
[legend]
```

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>close ()</td>
<td>Close the file.</td>
</tr>
<tr>
<td>bool</td>
<td>isEOF ()</td>
<td>Determines if the parser for this FileObject has reached the end of the file.</td>
</tr>
<tr>
<td>bool</td>
<td>openForAppend (string filename)</td>
<td>Open a specified file for writing, adding data to the end of the file.</td>
</tr>
<tr>
<td>bool</td>
<td>openForRead (string filename)</td>
<td>Open a specified file for reading.</td>
</tr>
<tr>
<td>bool</td>
<td>openForWrite (string filename)</td>
<td>Open a specified file for writing.</td>
</tr>
<tr>
<td>string</td>
<td>peekLine ()</td>
<td>Read a line from the file without moving the stream position.</td>
</tr>
<tr>
<td>string</td>
<td>readLine ()</td>
<td>Read a line from file.</td>
</tr>
<tr>
<td>void</td>
<td>writeLine (string text)</td>
<td>Write a line to the file, if it was opened for writing.</td>
</tr>
<tr>
<td>void</td>
<td>writeObject (SimObject *object)</td>
<td>Write an object to a text file.</td>
</tr>
<tr>
<td>void</td>
<td>writeObject (SimObject *object, string prepend)</td>
<td>Write an object to a text file, with some data added first.</td>
</tr>
</tbody>
</table>
Detailed Description

This class is responsible opening, reading, creating, and saving file contents.

FileObject acts as the interface with OS level files. You create a new FileObject and pass into it a file's path and name. The FileObject class supports three distinct operations for working with files:

<table>
<thead>
<tr>
<th>Operation</th>
<th>FileObject Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read</td>
<td>openForRead()</td>
<td>Open the file for reading</td>
</tr>
<tr>
<td>Write</td>
<td>openForWrite()</td>
<td>Open the file for writing to and replace its contents (if any)</td>
</tr>
<tr>
<td>Append</td>
<td>openForAppend()</td>
<td>Open the file and start writing at its end</td>
</tr>
</tbody>
</table>

Before you may work with a file you need to use one of the three above methods on the FileObject.

Example:

```java
// Create a file object for writing
%fileWrite = new FileObject();

// Open a file to write to, if it does not exist
%result = %fileWrite.OpenForWrite("./test.txt")
if ( %result )
{
    // Write a line to the text files
    %fileWrite.writeLine("READ. READ CODE.");
}

// Close the file when finished
%fileWrite.close();
```
// Cleanup the file object
%fileWrite.delete();

// Create a file object for reading
%fileRead = new FileObject();

// Open a text file, if it exists
%result = %fileRead.OpenForRead("./test.txt");

if ( %result )
{
    // Read in the first line
    %line = %fileRead.readline();

    // Print the line we just read
    echo(%line);
}

// Close the file when finished
%fileRead.close();

// Cleanup the file object
%fileRead.delete();
Member Function Documentation

void FileObject::close()

Close the file.

It is EXTREMELY important that you call this function when you are finished reading or writing to a file. Failing to do so is not only a bad programming practice, but could result in bad data or corrupt files. Remember: Open, Read/Write, Close, Delete...in that order!

Example:

```c++
// Create a file object for reading
%fileRead = new FileObject();

// Open a text file, if it exists
%fileRead.OpenForRead("./test.txt");

// Peek the first line
%line = %fileRead.peekLine();

// Print the line we just peeked
echo(%line);
// If we peek again...
%line = %fileRead.peekLine();

// We will get the same output as the first
// since the stream did not move forward
echo(%line);

// Close the file when finished
%fileWrite.close();
```
// Cleanup the file object
%fileWrite.delete();

bool FileObject::isEOF()

Determines if the parser for this FileObject has reached the end of the file.

Example:

// Create a file object for reading
%fileRead = new FileObject();

// Open a text file, if it exists
%fileRead.OpenForRead("./test.txt");

// Keep reading until we reach the end of
while( !%fileRead.isEOF() )
{
    %line = %fileRead.readline();
    echo(%line);
}

// Made it to the end
echo("Finished reading file");

Returns:
True if the parser has reached the end of the file, false otherwise

bool FileObject::openForAppend(string filename )
Open a specified file for writing, adding data to the end of the file.

There is no limit as to what kind of file you can write. Any format and data is allowable, not just text. Unlike `openForWrite()`, which will erase an existing file if it is opened, `openForAppend()` preserves data in an existing file and adds to it.

**Parameters:**

`filename Path`, name, and extension of file to append to

**Example:**

```c++
// Create a file object for writing
%fileWrite = new FileObject();

// Open a file to write to, if it does not
// If it does exist, whatever we write will
%result = %fileWrite.OpenForAppend("./test.txt")
```

**Returns:**

True if file was successfully opened, false otherwise

---

```c++
bool FileObject::openForRead(string filename )
```

Open a specified file for reading.

There is no limit as to what kind of file you can read. Any format and data contained within is accessible, not just text

**Parameters:**

`filename Path`, name, and extension of file to be read

**Example:**

```c++
// Create a file object for reading
```
%fileRead = new FileObject();
// Open a text file, if it exists
%result = %fileRead.OpenForRead("./test.txt")

**Returns:**
True if file was successfully opened, false otherwise

bool FileObject::openForWrite (string filename)

Open a specified file for writing.

There is no limit as to what kind of file you can write. Any format and data is allowable, not just text

**Parameters:**

*filename* Path, name, and extension of file to write to

**Example:**

// Create a file object for writing
%fileWrite = new FileObject();

// Open a file to write to, if it does not
%result = %fileWrite.OpenForWrite("./test.txt")

**Returns:**
True if file was successfully opened, false otherwise

string FileObject::peekLine()

Read a line from the file without moving the stream position.

Emphasis on *line*, as in you cannot parse individual characters
or chunks of data. There is no limitation as to what kind of data you can read. Unlike readLine, the parser does not move forward after reading.

**Parameters:**

`filename Path`, name, and extension of file to be read

**Example:**

```cpp
// Create a file object for reading
%fileRead = new FileObject();

// Open a text file, if it exists
%fileRead.OpenForRead("./test.txt");

// Peek the first line
%line = %fileRead.peekLine();

// Print the line we just peeked
echo(%line);

// If we peek again...
%line = %fileRead.peekLine();

// We will get the same output as the first since the stream did not move forward
echo(%line);
```

**Returns:**

String containing the line of data that was just peeked

```cpp
string FileObject::readLine() 
```

Read a line from file.
Emphasis on *line*, as in you cannot parse individual characters or chunks of data. There is no limitation as to what kind of data you can read.

Example:

```cpp
// Create a file object for reading
%fileRead = new FileObject();

// Open a text file, if it exists
%fileRead.OpenForRead("./test.txt");

// Read in the first line
%line = %fileRead.readline();

// Print the line we just read
echo(%line);
```

Returns:
String containing the line of data that was just read

```cpp
void FileObject::writeLine(string text )
```

Write a line to the file, if it was opened for writing.

There is no limit as to what kind of text you can write. Any format and data is allowable, not just text. Be careful of what you write, as whitespace, current values, and literals will be preserved.

Parameters:

text  The data we are writing out to file.

Example:

```cpp
// Create a file object for writing
%fileWrite = new FileObject();
```
// Open a file to write to, if it does not existWrite will be created
%fileWrite.OpenForWrite("./test.txt");

// Write a line to the text files
%fileWrite.writeLine("READ. READ CODE. CODE");

Returns:
True if file was successfully opened, false otherwise

void FileObject::writeObject(SimObject * object )

Write an object to a text file.

Unlike a simple writeLine using specified strings, this function writes an entire object to file, preserving its type, name, and properties. This is similar to the save() functionality of the SimObject class, but with a bit more control.

Parameters:

object The SimObject being written to file, properties, name, and all.

Example:

// Let's assume this SpawnSphere was created and exists in the running level
new SpawnSphere(TestSphere)
{
    spawnClass = "Player";
    spawnDatablock = "DefaultPlayerData";
    autoSpawn = "1";
    radius = "5";
    sphereWeight = "1";
    indoorWeight = "1";
}
outdoorWeight = "1";
dataBlock = "SpawnSphereMarker";
position = "-42.222 1.4845 4.80334";
rotation = "0 0 -1 108";
scale = "1 1 1";
canSaveDynamicFields = "1";
}

// Create a file object for writing
%fileWrite = new FileObject();

// Open a file to write to, if it does not exist it will be created
%fileWrite.OpenForWrite("./spawnSphers.txt");

// Write out the TestSphere
%fileWrite.writeObject(TestSphere);

// Close the text file
%fileWrite.close();

// Cleanup
%fileWrite.delete();
Parameters:

object The SimObject being written to file, properties, name, and all.

prepend Data or text that is written out before the SimObject.

Example:

```cpp
// Let's assume this SpawnSphere was created and currently exists in the running level
new SpawnSphere(TestSphere)
{
    spawnClass = "Player";
    spawnDatablock = "DefaultPlayerData";
    autoSpawn = "1";
    radius = "5";
    sphereWeight = "1";
    indoorWeight = "1";
    outdoorWeight = "1";
    dataBlock = "SpawnSphereMarker";
    position = 
```
// Close the text file
%fileWrite.close();

// Cleanup
%fileWrite.delete();
FileStreamObject Class Reference
[File I/O]

A wrapper around StreamObject for parsing text and data from files.
More...

Inheritance diagram for FileStreamObject:

List of all members.
**Public Member Functions**

<table>
<thead>
<tr>
<th>void</th>
<th>close ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Close the file. You can no longer read or write to it unless you open it again.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>open (string filename, string openMode)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Open a file for reading, writing, reading and writing, or appending.</td>
</tr>
</tbody>
</table>
Detailed Description

A wrapper around StreamObject for parsing text and data from files.

FileStreamObject inherits from StreamObject and provides some unique methods for working with strings. If you're looking for general file handling, you may want to use FileObject.

Example:

```plaintext
// Create a file stream object for reading
%fsObject = new FileStreamObject();

// Open a file for reading
%fsObject.open("./test.txt", "read");

// Get the status and print it
%status = %fsObject.getStatus();
echo(%status);

// Always remember to close a file stream
%fsObject.close();
```

See also:

StreamObject for the list of inherited functions variables
FileObject for general file handling.
**Member Function Documentation**

**void FileStreamObject::close()**

Close the file. You can no longer read or write to it unless you open it again.

**Example:**

```cpp
// Create a file stream object for reading
%fsObject = new FileStreamObject();

// Open a file for reading
%fsObject.open("./test.txt", "read");

// Always remember to close a file stream
%fsObject.close();
```

**See also:**

open()

**bool FileStreamObject::open(string filename, string openMode)**

Open a file for reading, writing, reading and writing, or appending.

Using "Read" for the open mode allows you to parse the contents of file, but not making modifications. "Write" will create a new file if it does not exist, or erase the contents of an existing file when opened. Write also allows you to modify the contents of the file.

"ReadWrite" will provide the ability to parse data (read it in) and manipulate data (write it out) interchangeably. Keep in mind the stream can move during each operation. Finally, "WriteAppend"
will open a file if it exists, but will not clear the contents. You can write new data starting at the end of the files existing contents.

**Parameters:**

- **filename**  Name of file to open
- **openMode** One of "Read", "Write", "ReadWrite" or "WriteAppend"

**Example:**

```csharp
// Create a file stream object for reading
%fsObject = new FileStreamObject();

// Open a file for reading
%fsObject.open("./test.txt", "read");

// Get the status and print it
%status = %fsObject.getStatus();
echo(%status);

// Always remember to close a file stream
%fsObject.close();
```

**Returns:**

True if the file was successfully opened, false if something went wrong

**See also:**

- close()
FlyingVehicle Class Reference
[Vehicles]

A flying vehicle. More...

Inheritance diagram for FlyingVehicle:

List of all members.
Public Member Functions

<table>
<thead>
<tr>
<th>void</th>
<th>useCreateHeight (bool enabled)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Set whether the vehicle should temporarily use the createHoverHeight specified in the datablock.</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

A flying vehicle.

The model used for the FlyingVehicle should contain the elements shown below. Only the collision mesh is actually required for the object to be added to the simulation, but particle emitters will not work unless the relevant nodes are present.

Collision mesh
- A convex collision mesh at detail size -1.

JetNozzle0-1 nodes
- Particle emitter nodes used when thrusting forward.

JetNozzle2-3 nodes
- Particle emitter nodes used when thrusting downward.

JetNozzleX, JetNozzleY nodes
- Particle emitter nodes used when thrusting backward.

contrail0-3 nodes
- Particle emitter nodes used when generating contrails.

activateBack animation
- Non-cyclic animation sequence played when the vehicle begins thrusting forwards.

maintainBack animation
- Cyclic animation sequence played after activateBack when the vehicle continues thrusting forwards.

activateBot animation
- Non-cyclic animation sequence played when the vehicle begins thrusting upwards.

maintainBot animation
- Cyclic animation sequence played after activateBot when the vehicle continues thrusting upwards.

The example below shows the datablock required for a simple FlyingVehicle. The script should be executed on the server, and the vehicle can then be added to the simulation programmatically from the level startup scripts, or by selecting the JetFighter datablock from
the World Editor (Library->ScriptedObjects->Vehicles).

Example:

```plaintext
datablock FlyingVehicleData( JetFighter ) {
    category = "Vehicles";
    shapeFile = "art/shapes/fighterjet.dae"
    createHoverHeight = 20;

    // 3rd person camera settings
    cameraRoll = true;
    cameraMaxDist = 16;
    cameraOffset = 1.0;
    cameraLag = 0.1;
    cameraDecay = 1.25;

    // Rigid Body
    mass = 100;
    massCenter = "0 -0.2 0";
    massBox = "0 0 0";
    integration = 3;
    collisionTol = 0.6;
    contactTol = 0.4;

    bodyFriction = 0;
    bodyRestitution = 0.8;
    minRollSpeed = 2000;
    minImpactSpeed = 5;
    softImpactSpeed = 3;
    hardImpactSpeed = 15;
    drag = 0.25;
}
```
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>minDrag</td>
<td>40;</td>
</tr>
<tr>
<td>rotationalDrag</td>
<td>20;</td>
</tr>
<tr>
<td>// Autostabilizer</td>
<td></td>
</tr>
<tr>
<td>maxAutoSpeed</td>
<td>6;</td>
</tr>
<tr>
<td>autoAngularForce</td>
<td>400;</td>
</tr>
<tr>
<td>autoLinearForce</td>
<td>300;</td>
</tr>
<tr>
<td>autoInputDamping</td>
<td>0.55;</td>
</tr>
<tr>
<td>// Maneuvering</td>
<td></td>
</tr>
<tr>
<td>maxSteeringAngle</td>
<td>3;</td>
</tr>
<tr>
<td>horizontalSurfaceForce</td>
<td>20;</td>
</tr>
<tr>
<td>verticalSurfaceForce</td>
<td>20;</td>
</tr>
<tr>
<td>maneuveringForce</td>
<td>6400;</td>
</tr>
<tr>
<td>steeringForce</td>
<td>500;</td>
</tr>
<tr>
<td>steeringRollForce</td>
<td>200;</td>
</tr>
<tr>
<td>rollForce</td>
<td>10;</td>
</tr>
<tr>
<td>hoverHeight</td>
<td>0.5;</td>
</tr>
<tr>
<td>createHoverHeight</td>
<td>0.5;</td>
</tr>
<tr>
<td>maxForwardSpeed</td>
<td>90;</td>
</tr>
<tr>
<td>// Vertical jetting</td>
<td></td>
</tr>
<tr>
<td>maxEnergy</td>
<td>100;</td>
</tr>
<tr>
<td>jetForce</td>
<td>3000;</td>
</tr>
<tr>
<td>minJetEnergy</td>
<td>28;</td>
</tr>
<tr>
<td>jetEnergyEnergyDrain</td>
<td>2.8;</td>
</tr>
<tr>
<td>vertThrustMultiple</td>
<td>3.0;</td>
</tr>
<tr>
<td>// Emitters</td>
<td></td>
</tr>
<tr>
<td>forwardJetEmitter</td>
<td>FighterJettingEmitter</td>
</tr>
<tr>
<td>backwardJetEmitter</td>
<td>FighterJettingEmitter</td>
</tr>
<tr>
<td>downJetEmitter</td>
<td>FighterJettingEmitter</td>
</tr>
<tr>
<td>trailEmitter</td>
<td>FighterContrail</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
minTrailSpeed = 10;

// Sounds
engineSound = FighterEngineSnd;
jetSound = FighterJettingSnd;
};

// This function is executed when the FlyingVehicle object is added to the simulation.
function JetFighter::onAdd( %this, %obj )
{
    Parent::onAdd( %this, %obj );

    // Allow jetting energy to recharge over time
    %obj.setRechargeRate( 2 );
}
void FlyingVehicle::useCreateHeight(bool enabled)

Set whether the vehicle should temporarily use the createHoverHeight specified in the datablock.

This can help avoid problems with spawning.

Parameters:

\[ \text{enabled} \]

true to use the datablock createHoverHeight, false otherwise
FlyingVehicleData Class Reference

[ Vehicles ]

Defines the properties of a FlyingVehicle. More...

Inheritance diagram for FlyingVehicleData:

- SimObject
- SimDataBlock
- GameBaseData
- ShapeBaseData
-VehicleData
-FlyingVehicleData

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>autoAngularForce</code></td>
<td>Corrective torque applied to level out the vehicle when moving at less than maxAutoSpeed.</td>
</tr>
<tr>
<td>float</td>
<td><code>autoInputDamping</code></td>
<td>Scale factor applied to steering input if speed is less than maxAutoSpeed to improve handling at very low speeds.</td>
</tr>
<tr>
<td>float</td>
<td><code>autoLinearForce</code></td>
<td>Corrective force applied to slow the vehicle when moving at less than maxAutoSpeed.</td>
</tr>
<tr>
<td><code>ParticleEmitterData</code></td>
<td><code>backwardJetEmitter</code></td>
<td>Emitter to generate particles for backward jet thrust.</td>
</tr>
<tr>
<td>float</td>
<td><code>createHoverHeight</code></td>
<td>The vehicle's height off the ground when useCreateHeight is active.</td>
</tr>
<tr>
<td><code>ParticleEmitterData</code></td>
<td><code>downJetEmitter</code></td>
<td>Emitter to generate particles for downward jet thrust.</td>
</tr>
<tr>
<td><code>SFXProfile</code></td>
<td><code>engineSound</code></td>
<td>Looping engine sound.</td>
</tr>
<tr>
<td><code>ParticleEmitterData</code></td>
<td><code>forwardJetEmitter</code></td>
<td>Emitter to generate particles for forward jet thrust.</td>
</tr>
<tr>
<td>float</td>
<td><code>horizontalSurfaceForce</code></td>
<td>Damping force in the opposite direction to sideways velocity.</td>
</tr>
<tr>
<td>float</td>
<td><code>hoverHeight</code></td>
<td>The vehicle's height off the ground when at</td>
</tr>
</tbody>
</table>
rest.

<table>
<thead>
<tr>
<th><strong>SFXProfile</strong></th>
<th><strong>jetSound</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Looping sound to play while the vehicle is jetting.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th><strong>maneuveringForce</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum X and Y (horizontal plane) maneuvering force.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th><strong>maxAutoSpeed</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum speed for automatic vehicle control assistance - vehicles travelling at speeds above this value do not get control assistance.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th><strong>minTrailSpeed</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum speed at which to start generating contrail particles.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th><strong>rollForce</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Damping torque against rolling maneuvers (rotation about the y-axis), proportional to linear velocity.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th><strong>rotationalDrag</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotational drag factor (slows vehicle rotation speed in all axes).</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th><strong>steeringForce</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum X and Z (sideways and vertical) steering force.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th><strong>steeringRollForce</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Roll force induced by sideways steering input value (controls how much the vehicle rolls when turning).</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>ParticleEmitterData</strong></th>
<th><strong>trailerEmitter</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Emitter to generate contrail particles from model nodes contrail0 - contrail3.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th><strong>verticalSurfaceForce</strong></th>
</tr>
</thead>
</table>
| Damping force in the opposite direction to
vertical velocity.

<table>
<thead>
<tr>
<th><strong>float</strong></th>
<th><strong>vertThrustMultiple</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiplier applied to the jetForce (defined in VehicleData) when thrusting vertically.</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

Defines the properties of a FlyingVehicle.
**Member Data Documentation**

```plaintext
float FlyingVehicleData::autoAngularForce

Corrective torque applied to level out the vehicle when moving at less than maxAutoSpeed.

The torque is inversely proportional to vehicle speed.

float FlyingVehicleData::autoInputDamping

Scale factor applied to steering input if speed is less than maxAutoSpeed to improve handling at very low speeds.

Smaller values make steering less sensitive.

float FlyingVehicleData::autoLinearForce

Corrective force applied to slow the vehicle when moving at less than maxAutoSpeed.

The force is inversely proportional to vehicle speed.

ParticleEmitterData FlyingVehicleData::backwardJetEmitter

Emitter to generate particles for backward jet thrust.

Backward jet thrust particles are emitted from model nodes JetNozzleX and JetNozzleY.

float FlyingVehicleData::createHoverHeight
```
The vehicle's height off the ground when useCreateHeight is active.

This can help avoid problems with spawning the vehicle.

**ParticleEmitterData FlyingVehicleData::downJetEmitter**

Emitter to generate particles for downward jet thrust.

Downward jet thrust particles are emitted from model nodes JetNozzle2 and JetNozzle3.

**SFXProfile FlyingVehicleData::engineSound**

Looping engine sound.

**ParticleEmitterData FlyingVehicleData::forwardJetEmitter**

Emitter to generate particles for forward jet thrust.

Forward jet thrust particles are emitted from model nodes JetNozzle0 and JetNozzle1.

**float FlyingVehicleData::horizontalSurfaceForce**

Damping force in the opposite direction to sideways velocity.

Provides "bite" into the wind for climbing/diving and turning).

**float FlyingVehicleData::hoverHeight**
The vehicle's height off the ground when at rest.

<table>
<thead>
<tr>
<th><strong>SFXProfile FlyingVehicleData::jetSound</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Looped sound to play while the vehicle is jetting.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float FlyingVehicleData::maneuveringForce</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum X and Y (horizontal plane) maneuvering force.</td>
</tr>
<tr>
<td>The actual force applied depends on the current thrust.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float FlyingVehicleData::maxAutoSpeed</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum speed for automatic vehicle control assistance - vehicles travelling at speeds above this value do not get control assistance.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float FlyingVehicleData::minTrailSpeed</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum speed at which to start generating contrail particles.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float FlyingVehicleData::rollForce</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Damping torque against rolling maneuvers (rotation about the y-axis), proportional to linear velocity.</td>
</tr>
<tr>
<td>Acts to adjust roll to a stable position over time as the vehicle moves.</td>
</tr>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td><strong>float FlyingVehicleData::rotationalDrag</strong></td>
</tr>
<tr>
<td><strong>float FlyingVehicleData::steeringForce</strong></td>
</tr>
<tr>
<td><strong>float FlyingVehicleData::steeringRollForce</strong></td>
</tr>
<tr>
<td><strong>ParticleEmitterData FlyingVehicleData::trailEmitter</strong></td>
</tr>
<tr>
<td><strong>float FlyingVehicleData::verticalSurfaceForce</strong></td>
</tr>
<tr>
<td><strong>float FlyingVehicleData::vertThrustMultiple</strong></td>
</tr>
</tbody>
</table>
Forest Class Reference
[Forest]

Forest is a global-bounds scene object provides collision and rendering for a (.forest) data file. More...

Inheritance diagram for Forest:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>void</th>
<th>clear ()</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>isDirty ()</td>
</tr>
<tr>
<td>void</td>
<td>regenCells ()</td>
</tr>
</tbody>
</table>
## Public Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filename</td>
<td><code>dataFile</code> The source forest data file.</td>
</tr>
<tr>
<td>bool</td>
<td><code>saveDataFile</code></td>
</tr>
<tr>
<td></td>
<td><code>saveDataFile(path)</code></td>
</tr>
</tbody>
</table>

### Lod

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>lodReflectScalar</code>             &lt;Scalar applied to the farclip distance when Forest renders into a reflection.&gt;`</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>disableImposters</code></td>
<td>A debugging aid which will disable rendering of all imposters in the forest.</td>
</tr>
<tr>
<td><code>drawBounds</code></td>
<td>A debugging aid which renders the forest bounds.</td>
</tr>
<tr>
<td><code>drawCells</code></td>
<td>A debugging aid which renders the forest cell bounds.</td>
</tr>
<tr>
<td><code>forceImposters</code></td>
<td>A debugging aid which will force all forest items to be rendered as imposters.</td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Forest is a global-bounds scene object provides collision and rendering for a (.forest) data file.

Forest is designed to efficiently render a large number of static meshes: trees, rocks plants, etc. These cannot be moved at game-time or play animations but do support wind effects using vertex shader transformations guided by vertex color in the asset and user placed wind emitters (or weapon explosions).

Script level manipulation of forest data is not possible through Forest, it is only the rendering/collision. All editing is done through the world editor.

See also:
  
  TSForestItemData Defines a tree type.  
  GuiForestEditorCtrl Used by the world editor to provide manipulation of forest data.
## Member Function Documentation

<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void Forest::clear( )</code></td>
</tr>
<tr>
<td><code>bool Forest::isDirty( )</code></td>
</tr>
<tr>
<td><code>void Forest::regenCells( )</code></td>
</tr>
</tbody>
</table>
Member Data Documentation

filename Forest::dataFile

The source forest data file.

float Forest::lodReflectScalar

Scalar applied to the farclip distance when Forest renders into a reflection.

bool Forest::saveDataFile

saveDataFile( [path] )
ForestBrushElement Class Reference
[Forest]

Represents a type of ForestItem and parameters for how it is placed when painting with a ForestBrush that contains it. More...

Inheritance diagram for ForestBrushElement:

```
  SimObject
   |
  -------
   |
ForestBrushElement
```

[legend]

List of all members.
## Public Attributes

### ForestBrushElement

*float* `elevationMax`  
The max world space elevation this item will be placed.

*float* `elevationMin`  
The min world space elevation this item will be placed.

### ForestItemData

- **ForestItemData**  
The type of ForestItem this element holds placement parameters for.

*float* `probability`  
The probability that this element will be created during an editor brush stroke is the sum of all element probabilities in the brush divided by the probability of this element.

*float* `rotationRange`  
The max rotation in degrees that items will be placed.

*float* `scaleExponent`  
An exponent used to bias between the minimum and maximum random sizes.

*float* `scaleMax`  
The maximum random size of each item.

*float* `scaleMin`  
The minimum random size for each item.

*float* `sinkMax`  
Max variation in the sink radius.

*float* `sinkMin`  
Min variation in the sink radius.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>float</strong> sinkRadius</td>
<td>This is the radius used to calculate how much to sink the trunk at its base and is used to sink the tree into the ground when its on a slope.</td>
</tr>
<tr>
<td><strong>float</strong> slopeMax</td>
<td>The max surface slope in degrees this item will be placed on.</td>
</tr>
<tr>
<td><strong>float</strong> slopeMin</td>
<td>The min surface slope in degrees this item will be placed on.</td>
</tr>
</tbody>
</table>
Detailed Description

Represents a type of ForestItem and parameters for how it is placed when painting with a ForestBrush that contains it.
Member Data Documentation

float ForestBrushElement::elevationMax

The max world space elevation this item will be placed.

float ForestBrushElement::elevationMin

The min world space elevation this item will be placed.

ForestItemData ForestBrushElement::ForestItemData

The type of ForestItem this element holds placement parameters for.

float ForestBrushElement::probability

The probability that this element will be created during an editor brush stroke is the sum of all element probabilities in the brush divided by the probability of this element.

float ForestBrushElement::rotationRange

The max rotation in degrees that items will be placed.

float ForestBrushElement::scaleExponent

An exponent used to bias between the minimum and maximum random sizes.
float ForestBrushElement::scaleMax

The maximum random size of each item.

float ForestBrushElement::scaleMin

The minimum random size for each item.

float ForestBrushElement::sinkMax

Max variation in the sink radius.

float ForestBrushElement::sinkMin

Min variation in the sink radius.

float ForestBrushElement::sinkRadius

This is the radius used to calculate how much to sink the trunk at its base and is used to sink the tree into the ground when its on a slope.

float ForestBrushElement::slopeMax

The max surface slope in degrees this item will be placed on.

float ForestBrushElement::slopeMin

The min surface slope in degrees this item will be placed on.
Copyright © GarageGames, LLC. All Rights Reserved.
ForestItemData Class Reference

[Forest]

Base class for defining a type of ForestItem. It does not implement loading or rendering of the shapeFile. More...

Inheritance diagram for ForestItemData:

List of all members.
## Public Attributes

### Wind

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>branchAmp</td>
<td>Amplitude of the effect on larger branches.</td>
</tr>
<tr>
<td>float</td>
<td>dampingCoefficient</td>
<td>Coefficient used in calculating spring forces on the trunk. Causes oscillation and forces to decay faster over time.</td>
</tr>
<tr>
<td>float</td>
<td>detailAmp</td>
<td>Amplitude of the winds effect on leafs/fronds.</td>
</tr>
<tr>
<td>float</td>
<td>detailFreq</td>
<td>Frequency (speed) of the effect on leafs/fronds.</td>
</tr>
<tr>
<td>float</td>
<td>mass</td>
<td>Mass used in calculating spring forces on the trunk. Generally how springy a plant is.</td>
</tr>
<tr>
<td>float</td>
<td>rigidity</td>
<td>Rigidity used in calculating spring forces on the trunk. How much the plant resists the wind force.</td>
</tr>
<tr>
<td>float</td>
<td>tightnessCoefficient</td>
<td>Coefficient used in calculating spring forces on the trunk. How much the plant resists bending.</td>
</tr>
<tr>
<td>float</td>
<td>trunkBendScale</td>
<td>Overall bend amount of the tree trunk by wind and impacts.</td>
</tr>
<tr>
<td>float</td>
<td>windScale</td>
<td>Overall scale to the effect of wind.</td>
</tr>
</tbody>
</table>

### Media

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>collidable</td>
<td>Can other objects or spacial queries hit items of this</td>
</tr>
<tr>
<td>Type</td>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>float</td>
<td>radius</td>
<td>Radius used during placement to ensure items are not crowded.</td>
</tr>
<tr>
<td>filename</td>
<td>shapeFile</td>
<td>Shape file for this item type.</td>
</tr>
</tbody>
</table>
Detailed Description

Base class for defining a type of ForestItem. It does not implement loading or rendering of the shapeFile.
## Member Data Documentation

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>ForestItemData::branchAmp</td>
<td>Amplitude of the effect on larger branches.</td>
</tr>
<tr>
<td>bool</td>
<td>ForestItemData::collidable</td>
<td>Can other objects or spacial queries hit items of this type.</td>
</tr>
<tr>
<td>float</td>
<td>ForestItemData::dampingCoefficient</td>
<td>Coefficient used in calculating spring forces on the trunk. Causes oscillation and forces to decay faster over time.</td>
</tr>
<tr>
<td>float</td>
<td>ForestItemData::detailAmp</td>
<td>Amplitude of the winds effect on leafs/fronds.</td>
</tr>
<tr>
<td>float</td>
<td>ForestItemData::detailFreq</td>
<td>Frequency (speed) of the effect on leafs/fronds.</td>
</tr>
<tr>
<td>float</td>
<td>ForestItemData::mass</td>
<td>Mass used in calculating spring forces on the trunk. Generally how springy a plant is.</td>
</tr>
<tr>
<td>float</td>
<td>ForestItemData::radius</td>
<td></td>
</tr>
</tbody>
</table>


Radius used during placement to ensure items are not crowded.

**float** *ForestItemData::rigidity*

Rigidity used in calculating spring forces on the trunk. How much the plant resists the wind force.

**filename** *ForestItemData::shapeFile*

Shape file for this item type.

**float** *ForestItemData::tightnessCoefficient*

Coefficient used in calculating spring forces on the trunk. How much the plant resists bending.

**float** *ForestItemData::trunkBendScale*

Overall bend amount of the tree trunk by wind and impacts.

**float** *ForestItemData::windScale*

Overall scale to the effect of wind.
ForestWindEmitter Class Reference
[Special Effects, Forest, Atmosphere]

Object responsible for simulating wind in a level. More...

Inheritance diagram for ForestWindEmitter:

```
SimObject
  
  NetObject
    
    SceneObject
      
      ForestWindEmitter
```

List of all members.
Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void attachToObject (int objectId)</td>
<td>Mounts the wind emitter to another scene object.</td>
</tr>
</tbody>
</table>
# Public Attributes

**ForestWind**

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>gustFrequency</code></td>
<td>The frequency of gusting in seconds.</td>
</tr>
<tr>
<td>float</td>
<td><code>gustStrength</code></td>
<td>The maximum strength of a gust.</td>
</tr>
<tr>
<td>float</td>
<td><code>gustWobbleStrength</code></td>
<td>The amount of random wobble added to gust and turbulence vectors.</td>
</tr>
<tr>
<td>float</td>
<td><code>gustYawAngle</code></td>
<td>The amount of degrees the wind direction can drift (both positive and negative).</td>
</tr>
<tr>
<td>float</td>
<td><code>gustYawFrequency</code></td>
<td>The frequency of wind yaw drift, in seconds.</td>
</tr>
<tr>
<td>bool</td>
<td><code>hasMount</code></td>
<td>Determines if the emitter is mounted to another object.</td>
</tr>
<tr>
<td>bool</td>
<td><code>radialEmitter</code></td>
<td>Determines if the emitter is a global direction or local radial emitter.</td>
</tr>
<tr>
<td>float</td>
<td><code>radius</code></td>
<td>The radius of the emitter for local radial emitters.</td>
</tr>
<tr>
<td>float</td>
<td><code>strength</code></td>
<td>The strength of the wind force.</td>
</tr>
<tr>
<td>float</td>
<td><code>turbulenceFrequency</code></td>
<td>The frequency of gust turbulence, in seconds.</td>
</tr>
<tr>
<td>float</td>
<td><code>turbulenceStrength</code></td>
<td>The strength of gust turbulence.</td>
</tr>
<tr>
<td>bool</td>
<td><code>windEnabled</code></td>
<td></td>
</tr>
</tbody>
</table>
Determines if the emitter will be counted in wind calculations.
## Static Public Attributes

<table>
<thead>
<tr>
<th>static bool</th>
<th>isRenderable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static bool</th>
<th>isSelectable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Object responsible for simulating wind in a level.

When placed in the level, a ForestWindEmitter will cause tree branches to bend and sway, leaves to flutter, and create vertical bending on the tree's trunk.

Example:

```plaintext
// The following is a full declaration of a ForestWindEmitter
new ForestWindEmitter()
{
    position = "497.739 765.821 102.395";
    windEnabled = "1";
    radialEmitter = "1";
    strength = "1";
    radius = "3";
    gustStrength = "0.5";
    gustFrequency = "1";
    gustYawAngle = "10";
    gustYawFrequency = "4";
    gustWobbleStrength = "2";
    turbulenceStrength = "1";
    turbulenceFrequency = "2";
    hasMount = "0";
    scale = "3 3 3";
    canSave = "1";
    canSaveDynamicFields = "1";
    rotation = "1 0 0 0";
};
```
Member Function Documentation

```cpp
void ForestWindEmitter::attachToObject(int objectID)
```

Mounts the wind emitter to another scene object.

**Parameters:**

- `objectID` Unique ID of the object wind emitter should attach to

**Example:**

```cpp
// Wind emitter previously created and named %windEmitter
// Going to attach it to the player, making him a walking wind storm
%windEmitter.attachToObject(%player);
```
### Member Data Documentation

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>gustFrequency</code></td>
<td>The frequency of gusting in seconds.</td>
</tr>
<tr>
<td>float</td>
<td><code>gustStrength</code></td>
<td>The maximum strength of a gust.</td>
</tr>
<tr>
<td>float</td>
<td><code>gustWobbleStrength</code></td>
<td>The amount of random wobble added to gust and turbulence vectors.</td>
</tr>
<tr>
<td>float</td>
<td><code>gustYawAngle</code></td>
<td>The amount of degrees the wind direction can drift (both positive and negative).</td>
</tr>
<tr>
<td>float</td>
<td><code>gustYawFrequency</code></td>
<td>The frequency of wind yaw drift, in seconds.</td>
</tr>
<tr>
<td>bool</td>
<td><code>hasMount</code></td>
<td>Determines if the emitter is mounted to another object.</td>
</tr>
<tr>
<td>bool</td>
<td><code>radialEmitter</code></td>
<td></td>
</tr>
</tbody>
</table>
Determines if the emitter is a global direction or local radial emitter.

```cpp
float ForestWindEmitter::radius
```

The radius of the emitter for local radial emitters.

```cpp
float ForestWindEmitter::strength
```

The strength of the wind force.

```cpp
float ForestWindEmitter::turbulenceFrequency
```

The frequency of gust turbulence, in seconds.

```cpp
float ForestWindEmitter::turbulenceStrength
```

The strength of gust turbulence.

```cpp
bool ForestWindEmitter::windEnabled
```

Determines if the emitter will be counted in wind calculations.
fxFoliageReplicator Class Reference

[Foliage]

An emitter to replicate fxFoliageItem objects across an area. More...

Inheritance diagram for fxFoliageReplicator:

List of all members.
Public Attributes

Restrictions

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int <code>AllowedTerrainSlope</code></td>
<td>Maximum surface angle allowed for foliage instances.</td>
</tr>
<tr>
<td>bool <code>AllowOnInteriors</code></td>
<td>Foliage will be placed on InteriorInstances when set.</td>
</tr>
<tr>
<td>bool <code>AllowOnStatics</code></td>
<td>Foliage will be placed on Static shapes when set.</td>
</tr>
<tr>
<td>bool <code>AllowOnTerrain</code></td>
<td>Foliage will be placed on terrain when set.</td>
</tr>
<tr>
<td>bool <code>AllowOnWater</code></td>
<td>Foliage will be placed on/under water when set.</td>
</tr>
<tr>
<td>bool <code>AllowWaterSurface</code></td>
<td>Foliage will be placed on water when set. Requires <code>AllowOnWater</code>.</td>
</tr>
</tbody>
</table>

Culling

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float <code>AlphaCutoff</code></td>
<td>Minimum alpha value allowed on foliage instances.</td>
</tr>
<tr>
<td>int <code>CullResolution</code></td>
<td>Minimum size of culling bins. Must be $\geq 8$ and $\leq$ OuterRadius.</td>
</tr>
<tr>
<td>float <code>FadeInRegion</code></td>
<td>Region beyond ViewDistance where foliage fades in/out.</td>
</tr>
<tr>
<td>float <code>FadeOutRegion</code></td>
<td>Region before ViewClosest where foliage fades in/out.</td>
</tr>
<tr>
<td>float <code>GroundAlpha</code></td>
<td>Alpha of the foliage at ground level. $0 = \text{transparent}$, $1 = \text{opaque}$</td>
</tr>
</tbody>
</table>
Use culling bins when enabled.

**ViewClosest**
Minimum distance from camera where foliage appears.

**ViewDistance**
Maximum distance from camera where foliage appears.

### Debugging

- **DebugBoxHeight**
  Height multiplier for drawn culling bins.

- **HideFoliage**
  Foliage is hidden when set to true.

- **PlacementAreaHeight**
  Height of the placement ring in world units.

- **PlacementColour**
  Color of the placement ring.

- **ShowPlacementArea**
  Draw placement rings when set to true.

- **UseDebugInfo**
  Culling bins are drawn when set to true.

### Dimensions

- **FixAspectRatio**
  Maintain aspect ratio of image if true. This option ignores MaxWidth.

- **FixSizeToMax**
  Use only MaxWidth and MaxHeight for billboard size. Ignores MinWidth and MinHeight.

- **MaxHeight**
<table>
<thead>
<tr>
<th><strong>Maximum height of foliage billboards.</strong></th>
</tr>
</thead>
</table>
| float **MaxWidth**  
Maximum width of foliage billboards. |
| float **MinHeight**  
Minimum height of foliage billboards. |
| float **MinWidth**  
Minimum width of foliage billboards. |
| float **OffsetZ**  
Offset billboards by this amount vertically. |
| bool **RandomFlip**  
Randomly flip billboards left-to-right. |
| bool **useTrueBillboards**  
Use camera facing billboards (including the z axis). |

**Media**

| int **FoliageCount**  
Maximum foliage instance count. |
| filename **FoliageFile**  
Image file for the foliage texture. |
| int **FoliageRetries**  
Number of times to try placing a foliage instance before giving up. |
| int **seed**  
Random seed for foliage placement. |

**Area**

| int **InnerRadiusX**  
Placement area inner radius on the X axis. |
| int **InnerRadiusY**  
Placement area inner radius on the Y axis. |
### OuterRadiusX
Placement area outer radius on the X axis.

### OuterRadiusY
Placement area outer radius on the Y axis.

### Lighting

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>LightOn</td>
<td>Foliage should be illuminated with changing lights when true.</td>
</tr>
<tr>
<td>bool</td>
<td>LightSync</td>
<td>Foliage instances have the same lighting when set and LightOn is set.</td>
</tr>
<tr>
<td>float</td>
<td>lightTime</td>
<td>Time before foliage illumination cycle repeats.</td>
</tr>
<tr>
<td>float</td>
<td>MaxLuminance</td>
<td>Maximum luminance for foliage instances.</td>
</tr>
<tr>
<td>float</td>
<td>MinLuminance</td>
<td>Minimum luminance for foliage instances.</td>
</tr>
</tbody>
</table>

### Animation

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>MaxSwayTime</td>
<td>Maximum sway cycle time in seconds.</td>
</tr>
<tr>
<td>float</td>
<td>MinSwayTime</td>
<td>Minimum sway cycle time in seconds.</td>
</tr>
<tr>
<td>float</td>
<td>SwayMagFront</td>
<td>Front-to-back sway magnitude.</td>
</tr>
<tr>
<td>float</td>
<td>SwayMagSide</td>
<td>Left-to-right sway magnitude.</td>
</tr>
<tr>
<td>bool</td>
<td>SwayOn</td>
<td>Foliage should sway randomly when true.</td>
</tr>
<tr>
<td>bool</td>
<td><strong>SwaySync</strong></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Foliage instances should sway together when true and SwayOn is enabled.</td>
<td></td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>static bool</th>
<th>isRenderable</th>
<th>Disables rendering of all instances of this type.</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

An emitter to replicate fxFoliageItem objects across an area.
## Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><code>fxFoliageReplicator::AllowedTerrainSlope</code></td>
</tr>
<tr>
<td>Maximum surface angle allowed for foliage instances.</td>
<td></td>
</tr>
<tr>
<td>bool</td>
<td><code>fxFoliageReplicator::AllowOnInteriors</code></td>
</tr>
<tr>
<td>Foliage will be placed on InteriorInstances when set.</td>
<td></td>
</tr>
<tr>
<td>bool</td>
<td><code>fxFoliageReplicator::AllowOnStatics</code></td>
</tr>
<tr>
<td>Foliage will be placed on Static shapes when set.</td>
<td></td>
</tr>
<tr>
<td>bool</td>
<td><code>fxFoliageReplicator::AllowOnTerrain</code></td>
</tr>
<tr>
<td>Foliage will be placed on terrain when set.</td>
<td></td>
</tr>
<tr>
<td>bool</td>
<td><code>fxFoliageReplicator::AllowOnWater</code></td>
</tr>
<tr>
<td>Foliage will be placed on/under water when set.</td>
<td></td>
</tr>
<tr>
<td>bool</td>
<td><code>fxFoliageReplicator::AllowWaterSurface</code></td>
</tr>
<tr>
<td>Foliage will be placed on water when set. Requires AllowOnWater.</td>
<td></td>
</tr>
<tr>
<td>float</td>
<td><code>fxFoliageReplicator::AlphaCutoff</code></td>
</tr>
<tr>
<td>Minimum alpha value allowed on foliage instances.</td>
<td></td>
</tr>
</tbody>
</table>
**int fxFoliageReplicator::CullResolution**

Minimum size of culling bins. Must be $\geq 8$ and $\leq$ OuterRadius.

**float fxFoliageReplicator::DebugBoxHeight**

Height multiplier for drawn culling bins.

**float fxFoliageReplicator::FadeInRegion**

Region beyond ViewDistance where foliage fades in/out.

**float fxFoliageReplicator::FadeOutRegion**

Region before ViewClosest where foliage fades in/out.

**bool fxFoliageReplicator::FixAspectRatio**

Maintain aspect ratio of image if true. This option ignores MaxWidth.

**bool fxFoliageReplicator::FixSizeToMax**

Use only MaxWidth and MaxHeight for billboard size. Ignores MinWidth and MinHeight.

**int fxFoliageReplicator::FoliageCount**
Maximum foliage instance count.

**filename fxFoliageReplicator::FoliageFile**

Image file for the foliage texture.

**int fxFoliageReplicator::FoliageRetries**

Number of times to try placing a foliage instance before giving up.

**float fxFoliageReplicator::GroundAlpha**

Alpha of the foliage at ground level. 0 = transparent, 1 = opaque.

**bool fxFoliageReplicator::HideFoliage**

Foliage is hidden when set to true.

**int fxFoliageReplicator::InnerRadiusX**

Placement area inner radius on the X axis.

**int fxFoliageReplicator::InnerRadiusY**

Placement area inner radius on the Y axis.

**bool fxFoliageReplicator::LightOn**
Foliage should be illuminated with changing lights when true.

**bool fxFoliageReplicator::LightSync**

Foliage instances have the same lighting when set and LightOn is set.

**float fxFoliageReplicator::lightTime**

Time before foliage illumination cycle repeats.

**float fxFoliageReplicator::MaxHeight**

Maximum height of foliage billboards.

**float fxFoliageReplicator::MaxLuminance**

Maximum luminance for foliage instances.

**float fxFoliageReplicator::MaxSwayTime**

Maximum sway cycle time in seconds.

**float fxFoliageReplicator::MaxWidth**

Maximum width of foliage billboards.

**float fxFoliageReplicator::MinHeight**
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float <code>fxFoliageReplicator::MinLuminance</code></td>
<td>Minimum luminance for foliage instances.</td>
</tr>
<tr>
<td>float <code>fxFoliageReplicator::MinSwayTime</code></td>
<td>Minimum sway cycle time in seconds.</td>
</tr>
<tr>
<td>float <code>fxFoliageReplicator::MinWidth</code></td>
<td>Minimum width of foliage billboards.</td>
</tr>
<tr>
<td>float <code>fxFoliageReplicator::OffsetZ</code></td>
<td>Offset billboards by this amount vertically.</td>
</tr>
<tr>
<td>int <code>fxFoliageReplicator::OuterRadiusX</code></td>
<td>Placement area outer radius on the X axis.</td>
</tr>
<tr>
<td>int <code>fxFoliageReplicator::OuterRadiusY</code></td>
<td>Placement area outer radius on the Y axis.</td>
</tr>
<tr>
<td>int <code>fxFoliageReplicator::PlacementAreaHeight</code></td>
<td></td>
</tr>
</tbody>
</table>
Height of the placement ring in world units.

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorF</td>
<td><code>fxFoliageReplicator::PlacementColour</code></td>
<td>Color of the placement ring.</td>
</tr>
<tr>
<td>bool</td>
<td><code>fxFoliageReplicator::RandomFlip</code></td>
<td>Randomly flip billboards left-to-right.</td>
</tr>
<tr>
<td>int</td>
<td><code>fxFoliageReplicator::seed</code></td>
<td>Random seed for foliage placement.</td>
</tr>
<tr>
<td>bool</td>
<td><code>fxFoliageReplicator::ShowPlacementArea</code></td>
<td>Draw placement rings when set to true.</td>
</tr>
<tr>
<td>float</td>
<td><code>fxFoliageReplicator::SwayMagFront</code></td>
<td>Front-to-back sway magnitude.</td>
</tr>
<tr>
<td>float</td>
<td><code>fxFoliageReplicator::SwayMagSide</code></td>
<td>Left-to-right sway magnitude.</td>
</tr>
<tr>
<td>bool</td>
<td><code>fxFoliageReplicator::SwayOn</code></td>
<td>Foliage should sway randomly when true.</td>
</tr>
</tbody>
</table>
bool fxFoliageReplicator::SwaySync

Foliage instances should sway together when true and SwayOn is enabled.

bool fxFoliageReplicator::UseCulling

Use culling bins when enabled.

bool fxFoliageReplicator::UseDebugInfo

Culling bins are drawn when set to true.

bool fxFoliageReplicator::useTrueBillboards

Use camera facing billboards (including the z axis).

float fxFoliageReplicator::ViewClosest

Minimum distance from camera where foliage appears.

float fxFoliageReplicator::ViewDistance

Maximum distance from camera where foliage appears.
fxShapeReplicatedStatic Class Reference
[Foliage]

The object definition for shapes that will be replicated across an area using an fxShapeReplicator. More...

Inheritance diagram for fxShapeReplicatedStatic:

List of all members.
## Public Attributes

### Collision

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>allowPlayerStep</code></td>
<td>Allow a Player to walk up sloping polygons in the TSStatic (based on the collisionType).</td>
</tr>
<tr>
<td>TSMeshType</td>
<td><code>collisionType</code></td>
<td>The type of mesh data to use for collision queries.</td>
</tr>
<tr>
<td>TSMeshType</td>
<td><code>decalType</code></td>
<td>The type of mesh data used to clip decal polygons against.</td>
</tr>
</tbody>
</table>

### Debug

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><code>forceDetail</code></td>
<td>Forces rendering to a particular detail level.</td>
</tr>
<tr>
<td>float</td>
<td><code>renderNormals</code></td>
<td>Debug rendering mode shows the normals for each point in the TSStatic's mesh.</td>
</tr>
</tbody>
</table>

### Rendering

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>meshCulling</code></td>
<td>Enables detailed culling of meshes within the TSStatic. Should only be used with large complex shapes like buildings which contain many submeshes.</td>
</tr>
<tr>
<td>bool</td>
<td><code>originSort</code></td>
<td>Enables translucent sorting of the TSStatic by its origin instead of the bounds.</td>
</tr>
<tr>
<td>bool</td>
<td><code>playAmbient</code></td>
<td>Enables automatic playing of the animation</td>
</tr>
</tbody>
</table>
sequence named "ambient" (if it exists) when the TSSStatic is loaded.

**Media**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filename</td>
<td>shapeName</td>
<td>Path and filename of the model file (.DTS, .DAE) to use for this TSSStatic.</td>
</tr>
<tr>
<td>string</td>
<td>skin</td>
<td>The skin applied to the shape.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

The object definition for shapes that will be replicated across an area using an fxShapeReplicator.
**Member Data Documentation**

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>bool fxShapeReplicatedStatic::allowPlayerStep</strong></td>
<td>Allow a Player to walk up sloping polygons in the TSStatic (based on the collisionType). When set to false, the slightest bump will stop the player from walking on top of the object.</td>
</tr>
<tr>
<td><strong>TSMeshType fxShapeReplicatedStatic::collisionType</strong></td>
<td>The type of mesh data to use for collision queries.</td>
</tr>
<tr>
<td><strong>TSMeshType fxShapeReplicatedStatic::decalType</strong></td>
<td>The type of mesh data used to clip decal polygons against.</td>
</tr>
<tr>
<td><strong>int fxShapeReplicatedStatic::forceDetail</strong></td>
<td>Forces rendering to a particular detail level.</td>
</tr>
<tr>
<td><strong>bool fxShapeReplicatedStatic::meshCulling</strong></td>
<td>Enables detailed culling of meshes within the TSStatic. Should only be used with large complex shapes like buildings which contain many submeshes.</td>
</tr>
<tr>
<td><strong>bool fxShapeReplicatedStatic::originSort</strong></td>
<td></td>
</tr>
</tbody>
</table>
Enables translucent sorting of the **TSSStatic** by its origin instead of the bounds.

```cpp
bool fxShapeReplicatedStatic::playAmbient
```

Enables automatic playing of the animation sequence named "ambient" (if it exists) when the **TSSStatic** is loaded.

```cpp
float fxShapeReplicatedStatic::renderNormals
```

Debug rendering mode shows the normals for each point in the TSSStatic's mesh.

```cpp
filename fxShapeReplicatedStatic::shapeName
```

Path and filename of the model file (.DTS, .DAE) to use for this **TSSStatic**.

```cpp
string fxShapeReplicatedStatic::skin
```

The skin applied to the shape.

'Skinning' the shape effectively renames the material targets, allowing different materials to be used on different instances of the same model.

Any material targets that start with the old skin name have that part of the name replaced with the new skin name. The initial old skin name is "base". For example, if a new skin of "blue" was applied to a model that had material targets **base_body** and **face**, the new targets would be **blue_body** and **face**. Note that **face** was not renamed since it did not start with the old skin name of "base".
To support models that do not use the default "base" naming convention, you can also specify the part of the name to replace in the skin field itself. For example, if a model had a material target called `shapemat`, we could apply a new skin "shape=blue", and the material target would be renamed to `bluemat` (note "shape" has been replaced with "blue").

Multiple skin updates can also be applied at the same time by separating them with a semicolon. For example: "base=blue;face=happy_face".

Material targets are only renamed if an existing Material maps to that name, or if there is a diffuse texture in the model folder with the same name as the new target.
**fxShapeReplicator Class Reference**

**[Foliage]**

An emitter for objects to replicate across an area. More...

Inheritance diagram for fxShapeReplicator:

```
SimObject

NetObject

SceneObject

fxShapeReplicator
```

List of all members.
## Public Attributes

### Restraints

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><strong>AlignToTerrain</strong></td>
<td>Align shapes to surface normal when set.</td>
</tr>
<tr>
<td>int</td>
<td><strong>AllowedTerrainSlope</strong></td>
<td>Maximum surface angle allowed for shape instances.</td>
</tr>
<tr>
<td>bool</td>
<td><strong>AllowOnInteriors</strong></td>
<td>Shapes will be placed on InteriorInstances when set.</td>
</tr>
<tr>
<td>bool</td>
<td><strong>AllowOnStatics</strong></td>
<td>Shapes will be placed on Static shapes when set.</td>
</tr>
<tr>
<td>bool</td>
<td><strong>AllowOnTerrain</strong></td>
<td>Shapes will be placed on terrain when set.</td>
</tr>
<tr>
<td>bool</td>
<td><strong>AllowOnWater</strong></td>
<td>Shapes will be placed on/under water when set.</td>
</tr>
<tr>
<td>bool</td>
<td><strong>AllowWaterSurface</strong></td>
<td>Shapes will be placed on water when set. Requires AllowOnWater.</td>
</tr>
<tr>
<td>Point3F</td>
<td><strong>TerrainAlignment</strong></td>
<td>Surface normals will be multiplied by these values when AlignToTerrain is enabled.</td>
</tr>
</tbody>
</table>

### Debugging

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><strong>HideReplications</strong></td>
<td>Replicated shapes are hidden when set to true.</td>
</tr>
<tr>
<td>int</td>
<td><strong>PlacementAreaHeight</strong></td>
<td>Height of the placement ring in world units.</td>
</tr>
</tbody>
</table>
**Color**

- **PlacementColour**
  Color of the placement ring.

**bool**

- **ShowPlacementArea**
  Draw placement rings when set to true.

## Placement Radius

- **int** **InnerRadiusX**
  Placement area inner radius on the X axis.

- **int** **InnerRadiusY**
  Placement area inner radius on the Y axis.

- **int** **OuterRadiusX**
  Placement area outer radius on the X axis.

- **int** **OuterRadiusY**
  Placement area outer radius on the Y axis.

## Object Transforms

- **int** **OffsetZ**
  Offset shapes by this amount vertically.

- **Point3F** **ShapeRotateMax**
  Maximum shape rotation angles.

- **Point3F** **ShapeRotateMin**
  Minimum shape rotation angles.

- **Point3F** **ShapeScaleMax**
  Maximum shape scale.

- **Point3F** **ShapeScaleMin**
  Minimum shape scale.

## Replications

- **int** **seed**
  Random seed for shape placement.
<table>
<thead>
<tr>
<th>int</th>
<th>ShapeCount</th>
<th>Maximum shape instance count.</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>ShapeRetries</td>
<td>Number of times to try placing a shape instance before giving up.</td>
</tr>
</tbody>
</table>

**Media**

| filename | shapeFile | Filename of shape to replicate. |
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

An emitter for objects to replicate across an area.
## Member Data Documentation

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bool fxShapeReplicator::AlignToTerrain</code></td>
<td>Align shapes to surface normal when set.</td>
</tr>
<tr>
<td><code>int fxShapeReplicator::AllowedTerrainSlope</code></td>
<td>Maximum surface angle allowed for shape instances.</td>
</tr>
<tr>
<td><code>bool fxShapeReplicator::AllowOnInteriors</code></td>
<td>Shapes will be placed on InteriorInstances when set.</td>
</tr>
<tr>
<td><code>bool fxShapeReplicator::AllowOnStatics</code></td>
<td>Shapes will be placed on Static shapes when set.</td>
</tr>
<tr>
<td><code>bool fxShapeReplicator::AllowOnTerrain</code></td>
<td>Shapes will be placed on terrain when set.</td>
</tr>
<tr>
<td><code>bool fxShapeReplicator::AllowOnWater</code></td>
<td>Shapes will be placed on/under water when set.</td>
</tr>
<tr>
<td><code>bool fxShapeReplicator::AllowWaterSurface</code></td>
<td>Shapes will be placed on water when set. Requires AllowOnWater.</td>
</tr>
</tbody>
</table>
bool fxShapeReplicator::HideReplications

Replicated shapes are hidden when set to true.

int fxShapeReplicator::InnerRadiusX

Placement area inner radius on the X axis.

int fxShapeReplicator::InnerRadiusY

Placement area inner radius on the Y axis.

bool fxShapeReplicator::Interactions

Allow physics interactions with shapes.

int fxShapeReplicator::OffsetZ

Offset shapes by this amount vertically.

int fxShapeReplicator::OuterRadiusX

Placement area outer radius on the X axis.

int fxShapeReplicator::OuterRadiusY

Placement area outer radius on the Y axis.
<p>| <strong>int fxShapeReplicator::PlacementAreaHeight</strong> | Height of the placement ring in world units. |
| <strong>ColorF fxShapeReplicator::PlacementColour</strong> | Color of the placement ring. |
| <strong>int fxShapeReplicator::seed</strong> | Random seed for shape placement. |
| <strong>int fxShapeReplicator::ShapeCount</strong> | Maximum shape instance count. |
| <strong>filename fxShapeReplicator::shapeFile</strong> | Filename of shape to replicate. |
| <strong>int fxShapeReplicator::ShapeRetries</strong> | Number of times to try placing a shape instance before giving up. |
| <strong>Point3F fxShapeReplicator::ShapeRotateMax</strong> | Maximum shape rotation angles. |</p>
<table>
<thead>
<tr>
<th><strong>Point3F fxShapeReplicator::ShapeRotateMin</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum shape rotation angles.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Point3F fxShapeReplicator::ShapeScaleMax</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum shape scale.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Point3F fxShapeReplicator::ShapeScaleMin</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum shape scale.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>bool fxShapeReplicator::ShowPlacementArea</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Draw placement rings when set to true.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Point3F fxShapeReplicator::TerrainAlignment</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Surface normals will be multiplied by these values when AlignToTerrain is enabled.</td>
</tr>
</tbody>
</table>
GameBase Class Reference
[Game Objects]

Base class for game objects which use datablocks, networking, are editable, and need to process ticks. More...

Inheritance diagram for GameBase:

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><strong>applyImpulse</strong> (Point3F pos, VectorF vel)</td>
<td>Apply an impulse to this object as defined by a world position and velocity vector.</td>
</tr>
<tr>
<td>void</td>
<td><strong>applyRadialImpulse</strong> (Point3F origin, float radius, float magnitude)</td>
<td>Applies a radial impulse to the object using the given origin and force.</td>
</tr>
<tr>
<td>int</td>
<td><strong>getDataBlock</strong> ()</td>
<td>Get the datablock used by this object.</td>
</tr>
<tr>
<td>bool</td>
<td><strong>setDataBlock</strong> (GameBaseData data)</td>
<td>Assign this GameBase to use the specified datablock.</td>
</tr>
</tbody>
</table>

### Callbacks

<table>
<thead>
<tr>
<th>Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td><strong>setControl</strong> (bool controlled)</td>
<td>Called when the client controlling the object changes.</td>
</tr>
</tbody>
</table>
**Public Attributes**

**Game**

| GameBaseData dataBlock | Script datablock used for game objects. |
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>boundingBox</td>
<td>Toggles on the rendering of the bounding boxes for certain types of objects in scene.</td>
</tr>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Base class for game objects which use datablocks, networking, are editable, and need to process ticks.
**Member Function Documentation**

```cpp
bool GameBase::applyImpulse (Point3F pos, 
VectorF vel )
```

Apply an impulse to this object as defined by a world position and velocity vector.

**Parameters:**
- `pos` impulse world position
- `vel` impulse velocity (impulse force $F = m \times v$)

**Returns:**
Always true

**Note:**
Not all objects that derive from `GameBase` have this defined.

```cpp
void GameBase::applyRadialImpulse (Point3F origin, 
float radius, 
float magnitude )
```

Applies a radial impulse to the object using the given origin and force.

**Parameters:**
- `origin` World point of origin of the radial impulse.
- `radius` The radius of the impulse area.
- `magnitude` The strength of the impulse.

**Note:**
Not all objects that derive from `GameBase` have this defined.
```cpp
int GameBase::getDataBlock()

Get the datablock used by this object.

**Returns:**

the datablock this `GameBase` is using.

**See also:**

`setDataBlock()`
```

```cpp
void GameBase::setControl(bool controlled)

Called when the client controlling the object changes.

**Parameters:**

`controlled` true if a client now controls this object, false if no client controls this object.
```

```cpp
bool GameBase::setDataBlock(GameBaseData data)

Assign this `GameBase` to use the specified datablock.

**Parameters:**

`data` new datablock to use

**Returns:**

true if successful, false if failed.

**See also:**

`getDataBlock()`
```
**Member Data Documentation**

**bool GameBase::boundingBox** [static]

Toggles on the rendering of the bounding boxes for certain types of objects in scene.

**GameBaseData GameBase::dataBlock**

Script datablock used for game objects.

Copyright © GarageGames, LLC. All Rights Reserved.
GameBaseData Class Reference
[Game Objects]

Scriptable, demo-able datablock. Used by GameBase objects.
More...

Inheritance diagram for GameBaseData:

List of all members.
**Public Member Functions**

**Callbacks**

<table>
<thead>
<tr>
<th>void</th>
<th>onAdd (GameBase obj)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Called when the object is added to the scene.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>onMount (GameBase obj, SceneObject mountObj, int node)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Called when the object is mounted to another object in the scene.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>onNewDataBlock (GameBase obj)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Called when the object has a new datablock assigned.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>onRemove (GameBase obj)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Called when the object is removed from the scene.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>onUnmount (GameBase obj, SceneObject mountObj, int node)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Called when the object is unmounted from another object in the scene.</td>
</tr>
</tbody>
</table>
Public Attributes

Scripting

caseString  category
The group that this datablock will show up in under the "Scripted" tab in the World Editor Library.
Detailed Description

Scriptable, demo-able datablock. Used by GameBase objects.

See also:

GameBase
Member Function Documentation

```c++
void GameBaseData::onAdd(GameBase obj)
```

Called when the object is added to the scene.

Parameters:

- `obj` the `GameBase` object

Example:

```c++
datablock GameBaseData(MyObjectData) {
    category = "Misc";
};

function MyObjectData::onAdd( %this, %obj {
    echo( "Added " @ %obj.getName() @ " to ");
}

function MyObjectData::onNewDataBlock( %this {
    echo( "Assign " @ %this.getName() @ ");
}

function MyObjectData::onRemove( %this, %obj {
    echo( "Removed " @ %obj.getName() @ " ");
}

function MyObjectData::onMount( %this, %obj, %mountObj, %node {
    echo( %obj.getName() @ " mounted to " @
```
void GameBaseData::onMount(GameBase obj, SceneObject mountObj, int node)

Called when the object is mounted to another object in the scene.

**Parameters:**

- *obj* the GameBase object being mounted
- *mountObj* the object we are mounted to
- *node* the mountObj node we are mounted to

**See also:**

onAdd for an example

void GameBaseData::onNewDataBlock(GameBase obj)

Called when the object has a new datablock assigned.

**Parameters:**

- *obj* the GameBase object

**See also:**

onAdd for an example
void GameBaseData::onRemove(GameBase obj)

Called when the object is removed from the scene.

**Parameters:**

- *obj* the GameBase object

**See also:**

- onAdd for an example

void GameBaseData::onUnmount(GameBase obj, SceneObject mountObj, int node)

Called when the object is unmounted from another object in the scene.

**Parameters:**

- *obj* the GameBase object being unmounted
- *mountObj* the object we are unmounted from
- *node* the mountObj node we are unmounted from

**See also:**

- onAdd for an example
Member Data Documentation

caseString GameBaseData::category

The group that this datablock will show up in under the "Scripted" tab in the World Editor Library.
GameConnection Class Reference
[Networking]

The game-specific subclass of NetConnection. More...

Inheritance diagram for GameConnection:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void activateGhosting()</code></td>
<td>Called by the server during phase 2 of the mission download to start sending ghosts to the client.</td>
</tr>
<tr>
<td><code>bool chaseCam (int size)</code></td>
<td>Sets the size of the chase camera's matrix queue.</td>
</tr>
<tr>
<td><code>void clearCameraObject()</code></td>
<td>Clear the connection's camera object reference.</td>
</tr>
<tr>
<td><code>void delete (string reason=&quot;&quot;)</code></td>
<td>On the server, disconnect a client and pass along an optional reason why.</td>
</tr>
<tr>
<td><code>SimObject getCameraObject()</code></td>
<td>Returns the connection's camera object used when not viewing through the control object.</td>
</tr>
<tr>
<td><code>float getControlCameraDefaultFov()</code></td>
<td>Returns the default field of view as used by the control object's camera.</td>
</tr>
<tr>
<td><code>float getControlCameraFov()</code></td>
<td>Returns the field of view as used by the control object's camera.</td>
</tr>
<tr>
<td><code>GameBase getControlObject()</code></td>
<td>On the server, returns the object that the client is controlling. By default the control object is an instance of the Player class, but can also be an instance of Camera (when editing the mission, for example), or any other ShapeBase derived class as appropriate for the game.</td>
</tr>
<tr>
<td><code>float getDamageFlash()</code></td>
<td>On the client, get the control object's damage flash level.</td>
</tr>
<tr>
<td><code>float getWhiteOut()</code></td>
<td></td>
</tr>
</tbody>
</table>
On the client, get the control object's white-out level.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool isAIControlled()</td>
<td>Returns true if this connection is AI controlled.</td>
</tr>
<tr>
<td>bool isControlObjectRotDampedCamera()</td>
<td>Returns true if the object being controlled by the client is making use of a rotation damped camera.</td>
</tr>
<tr>
<td>bool isDemoPlaying()</td>
<td>Returns true if a previously recorded demo file is now playing.</td>
</tr>
<tr>
<td>bool isDemoRecording()</td>
<td>Returns true if a demo file is now being recorded.</td>
</tr>
<tr>
<td>bool isFirstPerson()</td>
<td>Returns true if this connection is in first person mode.</td>
</tr>
<tr>
<td>void listClassIDs()</td>
<td>List all of the classes that this connection knows about, and what their IDs are. Useful for debugging network problems.</td>
</tr>
<tr>
<td>bool play2D (SFXProfile profile)</td>
<td>Used on the server to play a 2D sound that is not attached to any object.</td>
</tr>
<tr>
<td>bool play3D (SFXProfile profile, TransformF location)</td>
<td>Used on the server to play a 3D sound that is not attached to any object.</td>
</tr>
<tr>
<td>bool playDemo (string demoFileName)</td>
<td>On the client, play back a previously recorded game session.</td>
</tr>
<tr>
<td>void resetGhosting()</td>
<td>On the server, resets the connection to indicate that ghosting has been disabled.</td>
</tr>
<tr>
<td>void setBlackOut (bool doFade, int timeMS)</td>
<td>On the server, sets the client's 3D display to fade to black.</td>
</tr>
</tbody>
</table>
bool setCameraObject (GameBase camera)
On the server, set the connection's camera object used when not viewing through the control object.

void setConnectArgs (const char *args)
On the client, pass along a variable set of parameters to the server.

void setControlCameraFov (float newFOV)
On the server, sets the control object's camera's field of view.

bool setControlObject (GameBase ctrlObj)
On the server, sets the object that the client will control.

void setFirstPerson (bool firstPerson)
On the server, sets this connection into or out of first person mode.

void setJoinPassword (string password)
On the client, set the password that will be passed to the server.

void setMissionCRC (int CRC)
On the server, transmits the mission file's CRC value to the client.

void startRecording (string fileName)
On the client, starts recording the network connection's traffic to a demo file.

void stopRecording ()
On the client, stops the recording of a connection's network traffic to a file.

void transmitDataBlocks (int sequence)
Sent by the server during phase 1 of the mission download to send the datablocks to the client.

Callbacks
<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void initialControlSet ()</td>
<td>Called on the client when the first control object has been set by the server and we are now ready to go.</td>
</tr>
<tr>
<td>void onConnectionAccepted ()</td>
<td>Called on the client when the connection to the server has been established.</td>
</tr>
<tr>
<td>void onConnectionDropped (string reason)</td>
<td>Called on the client when the connection to the server has been dropped.</td>
</tr>
<tr>
<td>void onConnectionError (string errorString)</td>
<td>Called on the client when there is an error with the connection to the server.</td>
</tr>
<tr>
<td>void onConnectionTimedOut ()</td>
<td>Called on the client when the connection to the server times out.</td>
</tr>
<tr>
<td>void onConnectRequestRejected (string reason)</td>
<td>Called on the client when the connection to the server has been rejected.</td>
</tr>
<tr>
<td>void onConnectRequestTimedOut ()</td>
<td>Called when connection attempts have timed out.</td>
</tr>
<tr>
<td>void onControlObjectChange ()</td>
<td>Called on the client when the control object has been changed by the server.</td>
</tr>
<tr>
<td>void onDataBlocksDone (int sequence)</td>
<td>Called on the server when all datablocks has been sent to the client.</td>
</tr>
<tr>
<td>void onDrop (string disconnectReason)</td>
<td>Called on the server when the client's connection has been dropped.</td>
</tr>
<tr>
<td>void onFlash (bool state)</td>
<td>Called on the client when the damage flash or white out states change.</td>
</tr>
</tbody>
</table>
void setLagIcon (bool state)
    Called on the client to display the lag icon.
Static Public Member Functions

```
static int getServerConnection ()
On the client, this static method will return the connection to the server, if any.
```
Detailed Description

The game-specific subclass of NetConnection.

The GameConnection introduces the concept of the control object. The control object is simply the object that the client is associated with that network connection controls. By default the control object is an instance of the Player class, but can also be an instance of Camera (when editing the mission, for example), or any other ShapeBase derived class as appropriate for the game.

Torque uses a model in which the server is the authoritative master of the simulation. To prevent clients from cheating, the server simulates all player moves and then tells the client where his player is in the world. This model, while secure, can have problems. If the network latency is high, this round-trip time can give the player a very noticeable sense of movement lag. To correct this problem, the game uses a form of prediction - it simulates the movement of the control object on the client and on the server both. This way the client doesn't need to wait for round-trip verification of his moves. Only in the case of a force acting on the control object on the server that doesn't exist on the client does the client's position need to be forcefully changed.

To support this, all control objects (derivative of ShapeBase) must supply a writePacketData() and readPacketData() function that send enough data to accurately simulate the object on the client. These functions are only called for the current control object, and only when the server can determine that the client's simulation is somehow out of sync with the server. This occurs usually if the client is affected by a force not present on the server (like an interpolating object) or if the server object is affected by a server only force (such as the impulse from an explosion).

The Move structure is a 32 millisecond snapshot of player input, containing x, y, and z positional and rotational changes as well as trigger state changes. When time passes in the simulation moves are collected (depending on how much time passes), and applied to the current control object on the client. The same moves are then
packed over to the server in GameConnection::writePacket(), for processing on the server's version of the control object.

See also:

Networking, NetConnection, ShapeBase
Member Function Documentation

```cpp
void GameConnection::activateGhosting()
```

Called by the server during phase 2 of the mission download to start sending ghosts to the client.

Ghosts represent objects on the server that are in scope for the client. These need to be synchronized with the client in order for the client to see and interact with them. This is typically done during the standard mission start phase 2 when following Torque's example mission startup sequence.

Example:

```cpp
function serverCmdMissionStartPhase2Ack(%client, %seq, %playerDB)
{
    // Make sure to ignore calls from a previous mission load
    if (%seq != $missionSequence || !$MissionRunning)
        return;
    if (%client.currentPhase != 1.5)
        return;
    %client.currentPhase = 2;

    // Set the player datablock choice
    %client.playerDB = %playerDB;

    // Update mod paths, this needs to get there before the objects
    %client.transmitPaths();

    // Start ghosting objects to the client
    %client.activateGhosting();
}
```

See also:
On Ghosting and Scoping for a description of the ghosting system.

```cpp
bool GameConnection::chaseCam(int size)
```

Sets the size of the chase camera's matrix queue.

**Note:**

This sets the queue size across all GameConnections. This is not currently hooked up.

```cpp
void GameConnection::clearCameraObject()
```

Clear the connection's camera object reference.

**See also:**

GameConnection::setCameraObject() and GameConnection::getCameraObject()

```cpp
void GameConnection::delete(string reason = "")
```

On the server, disconnect a client and pass along an optional reason why.

This method performs two operations: it disconnects a client connection from the server, and it deletes the connection object. The optional reason is sent in the disconnect packet and is often displayed to the user so they know why they've been disconnected.

**Parameters:**

*reason* [optional] The reason why the user has been disconnected from the server.
Example:

```plaintext
function kick(%client)
{
  messageAll( 'MsgAdminForce', '\c2The Admin has kicked %1.'

  if (!%client.isAIControlled())
    BanList::add(%client.guid, %client.getAddress(), $Pref::Server::KickBanTime);

  %client.delete("You have been kicked from this server")
}
```

SimObject GameConnection::getCameraObject()

Returns the connection's camera object used when not viewing through the control object.

See also:

- GameConnection::setCameraObject()
- GameConnection::clearCameraObject()

float GameConnection::getControlCameraDefaultFov()

Returns the default field of view as used by the control object's camera.

float GameConnection::getControlCameraFov()

Returns the field of view as used by the control object's camera.

GameBase GameConnection::getControlObject()

On the server, returns the object that the client is controlling. By
default the control object is an instance of the \texttt{Player} class, but can also be an instance of \texttt{Camera} (when editing the mission, for example), or any other \texttt{ShapeBase} derived class as appropriate for the game.

\textbf{See also:}

\texttt{GameConnection::setControlObject()}

\begin{verbatim}
float GameConnection::getDamageFlash( )
\end{verbatim}

On the client, get the control object's damage flash level.

\textbf{Returns:}

\begin{itemize}
  \item flash level
\end{itemize}

\begin{verbatim}
static int GameConnection::getServerConnection( ) [static]
\end{verbatim}

On the client, this static method will return the connection to the server, if any.

\textbf{Returns:}

\begin{itemize}
  \item The \texttt{SimObject} ID of the server connection, or -1 if none is found.
\end{itemize}

\begin{verbatim}
float GameConnection::getWhiteOut( )
\end{verbatim}

On the client, get the control object's white-out level.

\textbf{Returns:}

\begin{itemize}
  \item white-out level
\end{itemize}

\begin{verbatim}
void GameConnection::initialControlSet( )
\end{verbatim}
Called on the client when the first control object has been set by the server and we are now ready to go.

A common action to perform when this callback is called is to switch the GUI canvas from the loading screen and over to the 3D game GUI.

```cpp
bool GameConnection::isAIControlled() {
    // Returns true if this connection is AI controlled.
}

See also:
AIConnection
```

```cpp
bool GameConnection::isControlObjectRotDampedCamera() {
    // Returns true if the object being controlled by the client is making use of a rotation damped camera.
}

See also:
Camera
```

```cpp
bool GameConnection::isDemoPlaying() {
    // Returns true if a previously recorded demo file is now playing.
}

See also:
GameConnection::playDemo()
```

```cpp
bool GameConnection::isDemoRecording() {
    // Returns true if a demo file is now being recorded.
}
See also:

GameConnection::startRecording(),
GameConnection::stopRecording()

bool GameConnection::isFirstPerson( )

Returns true if this connection is in first person mode.

Note:

Transition to first person occurs over time via mCameraPos, so this won't immediately return true after a set.

void GameConnection::listClassIDs( )

List all of the classes that this connection knows about, and what their IDs are. Useful for debugging network problems.

Note:

The list is sent to the console.

void GameConnection::onConnectionAccepted( )

Called on the client when the connection to the server has been established.

void GameConnection::onConnectionDropped(string reason)

Called on the client when the connection to the server has been dropped.

Parameters:

  reason  The reason why the connection was dropped.
void GameConnection::onConnectionError (string errorString )

Called on the client when there is an error with the connection to the server.

Parameters:

   errorString The connection error text.


void GameConnection::onConnectionTimedOut ( )

Called on the client when the connection to the server times out.


void GameConnection::onConnectRequestRejected (string reason )

Called on the client when the connection to the server has been rejected.

Parameters:

   reason The reason why the connection request was rejected.


void GameConnection::onConnectRequestTimedOut ( )

Called when connection attempts have timed out.


void GameConnection::onControlObjectChange ( )

Called on the client when the control object has been changed by the server.
void GameConnection::onDataBlocksDone(int sequence )

Called on the server when all datablocks has been sent to the client.

During phase 1 of the mission download, all datablocks are sent from the server to the client. Once all datablocks have been sent, this callback is called and the mission download procedure may move on to the next phase.

Parameters:

the sequence is common between the server and client and ensures that the client is acting on the most recent mission start process. If an errant network packet (one that was lost but has now been found) is received by the client with an incorrect sequence, it is just ignored. This sequence number is updated on the server every time a mission is loaded.

See also:

GameConnection::transmitDataBlocks()

void GameConnection::onDrop(string disconnectReason )

Called on the server when the client's connection has been dropped.

Parameters:

the disconnectReason is the reason why the connection was dropped.

void GameConnection::onFlash(bool state )

Called on the client when the damage flash or white out states
change.

When the server changes the damage flash or white out values, this callback is called either is on or both are off. Typically this is used to enable the flash postFx.

**Parameters:**
- `state` Set to true if either the damage flash or white out conditions are active.

```cpp
bool GameConnection::play2D(SFXProfile profile)
```

Used on the server to play a 2D sound that is not attached to any object.

**Parameters:**
- `profile` The `SFXProfile` that defines the sound to play.

**Example:**

```cpp
function ServerPlay2D(%profile)
{
    // Play the given sound profile on every client.
    // The sounds will be transmitted as an event, not attached to any object.
    for(%idx = 0; %idx < ClientGroup.getCount(); %idx++)
    {
        ClientGroup.getObject(%idx).play2D(%profile);
    }
}
```

```cpp
bool GameConnection::play3D(SFXProfile profile, TransformF location)
```

Used on the server to play a 3D sound that is not attached to any object.
Parameters:

- **profile**  The SFXProfile that defines the sound to play.
- **location** The position and orientation of the 3D sound given in the form of "x y z ax ay az aa".

Example:

```plaintext
function ServerPlay3D(%profile, %transform) {
    // Play the given sound profile at the
    // The sound will be transmitted as an
    for (%idx = 0; %idx < ClientGroup.getCount(); %idx++)
        ClientGroup.getObject(%idx).play3D(%profile, %transform);
}
```

```plaintext
bool GameConnection::playDemo(string demoFileName )
```

On the client, play back a previously recorded game session.

It is often useful to play back a game session. This could be for producing a demo of the game that will be shown at a later time, or for debugging a game. By recording the entire network stream it is possible to later play game the game exactly as it unfolded during the actual play session. This is because all user control and server results pass through the connection.

**Returns:**

True if the playback was successful. False if there was an issue, such as not being able to open the demo file for playback.

**See also:**

- `GameConnection::startRecording()`
- `GameConnection::stopRecording()`
void GameConnection::resetGhosting()

On the server, resets the connection to indicate that ghosting has been disabled.

Typically when a mission has ended on the server, all connected clients are informed of this change and their connections are reset back to a starting state. This method resets a connection on the server to indicate that ghosts are no longer being transmitted. On the client end, all ghost information will be deleted.

Example:

```cpp
// Inform the clients
for (%clientIndex = 0; %clientIndex < ClientGroup.getCount(); %clientIndex++)
{
    // clear ghosts and paths from all clients
    %cl = ClientGroup.getObject(%clientIndex);
    %cl.endMission();
    %cl.resetGhosting();
    %cl.clearPaths();
}
```

See also:

On Ghosting and Scoping for a description of the ghosting system.

void GameConnection::setBlackOut(bool doFade, int timeMS)

On the server, sets the client's 3D display to fade to black.

**Parameters:**

Set to true to fade to black, and false to fade from
doFade black.

timeMS Time it takes to perform the fade as measured in ms.

Note:
Not currently hooked up, and is not synchronized over the network.

```cpp
bool GameConnection::setCameraObject(GameBase camera)
```

On the server, set the connection's camera object used when not viewing through the control object.

See also:
- GameConnection::getCameraObject()
- GameConnection::clearCameraObject()

```cpp
void GameConnection::setConnectArgs(const char * args)
```

On the client, pass along a variable set of parameters to the server.

Once the connection is established with the server, the server calls its onConnect() method with the client's passed in parameters as arguments.

See also:
- GameConnection::onConnect()

```cpp
void GameConnection::setControlCameraFov(float newFOV)
```

On the server, sets the control object's camera's field of view.

Parameters:
New field of view (in degrees) to force the control object's camera to use. This value is clamped to be within the range of 1 to 179 degrees.

**Note:**

When transmitted over the network to the client, the resolution is limited to one degree. Any fraction is dropped.

```cpp
bool GameConnection::setControlObject(GameBase ctrlObj)
```

On the server, sets the object that the client will control.

By default the control object is an instance of the `Player` class, but can also be an instance of `Camera` (when editing the mission, for example), or any other `ShapeBase` derived class as appropriate for the game.

**Parameters:**

- `ctrlObj` The `GameBase` object on the server to control.

```cpp
void GameConnection::setFirstPerson(bool firstPerson)
```

On the server, sets this connection into or out of first person mode.

**Parameters:**

- `firstPerson` Set to true to put the connection into first person mode.

```cpp
void GameConnection::setJoinPassword(string password)
```

On the client, set the password that will be passed to the server.

On the server, this password is compared with what is stored in
pref

:Server::Password. If

pref::Server::Password is empty then the client's sent password is ignored. Otherwise, if the passed in client password and the server password do not match, the CHR_PASSWORD error string is sent back to the client and the connection is immediately terminated.

This password checking is performed quite early on in the connection request process so as to minimize the impact of multiple failed attempts -- also known as hacking.

void GameConnection::setLagIcon(bool state)

Called on the client to display the lag icon.

When the connection with the server is lagging, this callback is called to allow the game GUI to display some indicator to the player.

Parameters:

  state  Set to true if the lag icon should be displayed.

void GameConnection::setMissionCRC(int CRC)

On the server, transmits the mission file's CRC value to the client.

Typically, during the standard mission start phase 1, the mission file's CRC value on the server is send to the client. This allows the client to determine if the mission has changed since the last time it downloaded this mission and act appropriately, such as rebuilt cached lightmaps.

Parameters:

  CRC  The mission file's CRC value on the server.
Example:

```c
function serverCmdMissionStartPhase1Ack(%client, %seq)
{
    // Make sure to ignore calls from a previous mission
    if (%seq != $missionSequence || !$MissionRunning)
        return;
    if (%client.currentPhase != 0)
        return;
    %client.currentPhase = 1;
    // Start with the CRC
    %client.setMissionCRC($missionCRC);
    // Send over the datablocks...
    // OnDataBlocksDone will get called when they've all been received.
    %client.transmitDataBlocks($missionSequence);
}
```

```c
void GameConnection::startRecording(string fileName)
```

On the client, starts recording the network connection's traffic to a demo file.

It is often useful to play back a game session. This could be for producing a demo of the game that will be shown at a later time, or for debugging a game. By recording the entire network stream it is possible to later play game the game exactly as it unfolded during the actual play session. This is because all user control and server results pass through the connection.

**Parameters:**

- `fileName` The file name to use for the demo recording.
See also:

GameConnection::stopRecording(),
GameConnection::playDemo()

void GameConnection::stopRecording(
)

On the client, stops the recording of a connection's network traffic to a file.

See also:

GameConnection::startRecording(),
GameConnection::playDemo()

void GameConnection::transmitDataBlocks(int sequence)

Sent by the server during phase 1 of the mission download to send the datablocks to the client.

SimDataBlocks, also known as just datablocks, need to be transmitted to the client prior to the client entering the game world. These represent the static data that most objects in the world reference. This is typically done during the standard mission start phase 1 when following Torque's example mission startup sequence.

When the datablocks have all been transmitted, onDataBlocksDone() is called to move the mission start process to the next phase.

Parameters:

The sequence is common between the server and client and ensures that the client is acting on the most recent mission start process. If an errant network packet (one that was lost but has now been found) is received by the client with an
incorrect sequence, it is just ignored. This sequence number is updated on the server every time a mission is loaded.

Example:

```c
function serverCmdMissionStartPhase1Ack(%client,%seq) {
    // Make sure to ignore calls from a previous mission load
    if (%seq != $missionSequence || !$MissionRunning)
        return;
    if (%client.currentPhase != 0)
        return;
    %client.currentPhase = 1;

    // Start with the CRC
    %client.setMissionCRC( $missionCRC );

    // Send over the datablocks...
    // OnDataBlocksDone will get called when
    // that they've all been received.
    %client.transmitDataBlocks($missionSequence);
}
```

See also:

`GameConnection::onDataBlocksDone()`
GameTSCtrl Class Reference
[3D Controls]

The main 3D viewport for a Torque 3D game. More...

Inheritance diagram for GameTSCtrl:

- SimObject
  - SimSet
  - SimGroup
  - GuiControl
  - GuiContainer
  - GuiTSCtrl
  - GameTSCtrl [legend]

List of all members.
**Detailed Description**

The main 3D viewport for a Torque 3D game.

With the exception of a few very niche genres, the bulk of your 3D game viewing will occur in a GameTSCtrl. You typically only need a single GameTSCtrl, unless you are implementing a very complex interface system. In the demos, you can find our example named "PlayGui".

It is recommended that any game GUIs that are not pushed and popped constantly, be contained within your GameTSCtrl. Examples include targeting reticle, standard healthbar, ammo count, etc. This is mostly a design decision, but the way Torque 3D's GUI system works somewhat encourages you to group the controls in this manner.

```csharp
// Example of a GameTSCtrl
// PlayGui is the main TSControl through which the game is viewed
// Also contains a Guis for:
// - A lag icon
// - Showing other shape names
// - Crossahir

%guiContent = new GameTSCtrl(PlayGui)
{
    cameraZRot = "0"
    forceFOV = "0"
    reflectPriority = "1"
    Profile = "GuiContentProfile"
    HorizSizing = "right"
    VertSizing = "bottom"
    position = "0 0"
    Extent = "1024 768"

    new GuiBitmapCtrl(LagIcon)
}```
new GuiShapeNameHud()
{
    fillColor = "0 0 0 0.25"
    frameColor = "0 1 0 1"
    textColor = "0 1 0 1"
    showFill = "0"
    showFrame = "0"

    // Note: Rest of fields hidden for this example
};

new GuiCrossHairHud(Reticle)
{
    damageFillColor = "0 1 0 1"
    damageFrameColor = "1 0.6 0 1"
    damageRect = "50 4"
    damageOffset = "0 10"
    bitmap = "art/gui/weaponHud/blank.png"

    // Note: Rest of fields hidden for this example
};

See also:
GuiTSCtrl
GuiObjectView
GFXCardProfiler Class Reference

Provider a device independent wrapper around both the capabilities reported by the card/drivers and the exceptions recorded in various scripts. More...
Detailed Description

Provides a device independent wrapper around both the capabilities reported by the card/drivers and the exceptions recorded in various scripts.

The GFXCardProfiler provides a device independent wrapper around both the capabilities reported by the card/drivers and the exceptions recorded in various scripts.

The materials system keeps track of most caps-related rendering optimizations and/or workarounds, but it is occasionally necessary to expose capability information to higher level code (for instance, if some feature depends on a specific subset of render functionality) or to keep track of exceptions.

The proper way to fix this is to get the IHV to release fixed drivers and/or move to a single consistent rendering path that works. Of course, when you're releasing a game, especially on a timeline (or with a less than infinite budget) this isn't always a valid solution.

It's also often convenient to be able to tweak performance/detail settings based on the identified card type.

GFXCardProfiler addresses both these needs by providing two data retrieval methods and a generic interface for querying capability strings.

Note:

The GFXCardProfiler is at heart a system for implementing WORKAROUNDS. It is not guaranteed to work in all cases. The capability strings it responds to are specific to each implementation. You should be EXTREMELY careful when working with this functionality. When used in moderation it can be a project-saver, but if used to excess or without forethought it can lead to complex, hard-to-maintain code.

The first data retrieval method that the GFXCardProfiler supports is a card-specific capability query. This is implemented by each subclass.
In the case of DirectX, this means using the built-in capability query. For OpenGL or other APIs, more exotic methods may be necessary. The goal of this method is to retrieve some reasonable defaults that can be overridden later if necessary.

The second data retrieval method is script based. In ./profile a collection of script files are stored. They are named in one of the forms:

- **Renderer.cs**

These files are found and executed from most general to most specific. For instance, say we're working in the D3D renderer with an nVidia GeForce FX 5950, running driver version 53.36. The following files would be found and executed:

- **D3D.cs**
- **D3D.nVidia.cs**
- **D3D.nVidia.GeForceFX5950.cs**
- **D3D.nVidia.GeForceFX5950.5336.cs**

The general rule for turning strings into filename parts is to strip all spaces and punctuation. If a file is not found, no error is reported; it is assumed that the absence of a file means all is well.

Several functions are made available to allow simple logic in the script functions (for instance, to enable a workaround for a given range of driver versions). They are:

- GFXCardProfiler::getRenderer()
- GFXCardProfiler::getVendor()
- GFXCardProfiler::getCard()
- GFXCardProfiler::getVersion()
In addition, specific subclasses may expose other values (for instance, chipset IDs). These are made available as static members of the specific subclass. For instance, a D3D-specific chipset query may be made available as GFXD3DCardProfiler::getChipset()

Finally, once a script file has reached a determination they may indicate their settings to the GFXCardProfiler by calling GFXCardProfiler::setCapability(). For instance,

```
// Indicate we can show the color red.
GFXCardProfiler::setCapability("supportsRed")
```

GFXCardProfiler may be queried from script by calling GFXCardProfiler::queryProfile() - for instance:

```
GFXCardProfiler::queryProfile("supportsRed")
```
GFXCardProfilerAPI Class Reference

This class is the interface between TorqueScript and GFXCardProfiler. More...

List of all members.
## Static Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static String getCard ()</td>
<td>Returns the card name.</td>
</tr>
<tr>
<td>static String getRenderer ()</td>
<td>Returns the renderer name. For example D3D9 or OpenGL.</td>
</tr>
<tr>
<td>static String getVendor ()</td>
<td>Returns the card vendor name.</td>
</tr>
<tr>
<td>static String getVersion ()</td>
<td>Returns the driver version string.</td>
</tr>
<tr>
<td>static int getVideoMemoryMB ()</td>
<td>Returns the amount of video memory in megabytes.</td>
</tr>
<tr>
<td>static int queryProfile (string name, int defaultValue)</td>
<td>Used to query the value of a specific card capability.</td>
</tr>
<tr>
<td>static void setCapability (string name, int value)</td>
<td>Used to set the value for a specific card capability.</td>
</tr>
</tbody>
</table>
Detailed Description

This class is the interface between TorqueScript and GFXCardProfiler.

You will not actually declare GFXCardProfilerAPI in TorqueScript. It exists solely to give access to the GFXCardProfiler's querying functions, such as GFXCardProfiler::getRenderer.

Example:

```cpp
// Example of accessing GFXCardProfiler function
// Notice you are not using the API version
%videoMem = GFXCardProfiler::getVideoMemoryMB
```

See also:

GFXCardProfiler for more information
Member Function Documentation

**static String GFXCardProfilerAPI::getCard( ) [static]**

Returns the card name.

**static String GFXCardProfilerAPI::getRenderer( ) [static]**

Returns the renderer name. For example D3D9 or OpenGL.

**static String GFXCardProfilerAPI::getVendor( ) [static]**

Returns the card vendor name.

**static String GFXCardProfilerAPI::getVersion( ) [static]**

Returns the driver version string.

**static int GFXCardProfilerAPI::getVideoMemoryMB( ) [static]**

Returns the amount of video memory in megabytes.

**static int GFXCardProfilerAPI::queryProfile(string name, int defaultValue ) [static]**

Used to query the value of a specific card capability.

**Parameters:**

- *name* The name of the capability being queried.
defaultValue  The value to return if the capability is not defined.

```cpp
static void GFXCardProfilerAPI::setCapability(string name, int value) [static]
```

Used to set the value for a specific card capability.

**Parameters:**

- *name*  The name of the capability being set.
- *value*  The value to set for that capability.

Copyright © GarageGames, LLC. All Rights Reserved.
GFXInit Class Reference

[GFX]

Functions for tracking GFX adapters and initializing them into devices. More...

List of all members.
### Static Public Member Functions

<table>
<thead>
<tr>
<th>Method Type</th>
<th>Method Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static void</td>
<td><code>createNullDevice</code> ()</td>
<td>Create the NULL graphics device used for testing or headless operation.</td>
</tr>
<tr>
<td>static int</td>
<td><code>getAdapterCount</code> ()</td>
<td>Return the number of graphics adapters available.</td>
</tr>
<tr>
<td>static String</td>
<td><code>getAdapterMode</code> (int index, int modeIndex)</td>
<td>Gets the details of the specified adapter mode.</td>
</tr>
<tr>
<td>static int</td>
<td><code>getAdapterModeCount</code> (int index)</td>
<td>Gets the number of modes available on the specified adapter.</td>
</tr>
<tr>
<td>static String</td>
<td><code>getAdapterName</code> (int index)</td>
<td>Returns the name of the graphics adapter.</td>
</tr>
<tr>
<td>static float</td>
<td><code>getAdapterShaderModel</code> (int index)</td>
<td>Returns the supported shader model of the graphics adapter or -1 if the index is bad.</td>
</tr>
<tr>
<td>static GFXAdapterType</td>
<td><code>getAdapterType</code> (int index)</td>
<td>Returns the type (D3D9, D3D8, GL, Null) of a graphics adapter.</td>
</tr>
<tr>
<td>static int</td>
<td><code>getDefaultAdapterIndex</code> ()</td>
<td>Returns the index of the default graphics adapter. This is the graphics device which will be used to initialize the engine.</td>
</tr>
</tbody>
</table>
Detailed Description

Functions for tracking GFX adapters and initializing them into devices.
Member Function Documentation

static void GFXInit::createNullDevice() [static]

Create the NULL graphics device used for testing or headless operation.

static String GFXInit::getAdapterMode(int index, int modeIndex) [static]

Gets the details of the specified adapter mode.

Parameters:

- **index**: Index of the adapter to query.
- **modeIndex**: Index of the mode to get data from.

Returns:

A video mode string in the format 'width height fullscreen bitDepth refreshRate aaLevel'.

See also:

- GuiCanvas::getVideoMode()

static int GFXInit::getAdapterModeCount(int index) [static]

Gets the number of modes available on the specified adapter.

Parameters:

- **index**: Index of the adapter to get modes from.

Returns:

The number of video modes supported by the adapter or -1 if
the given adapter was not found.

```
static String GFXInit::getAdapterName(int index ) [static]
```

Returns the name of the graphics adapter.

**Parameters:**

- `index` The index of the adapter.

```
static float GFXInit::getAdapterShaderModel(int index ) [static]
```

Returns the supported shader model of the graphics adapter or -1 if the index is bad.

**Parameters:**

- `index` The index of the adapter.

```
static GFXAdapterType GFXInit::getAdapterType(int index ) [static]
```

Returns the type (D3D9, D3D8, GL, Null) of a graphics adapter.

**Parameters:**

- `index` The index of the adapter.

```
static int GFXInit::getDefaultAdapterIndex( ) [static]
```

Returns the index of the default graphics adapter. This is the graphics device which will be used to initialize the engine.
GFXSamplerStateData Class Reference

A sampler state used by GFXStateBlockData. More...

Inheritance diagram for GFXSamplerStateData:

```
Legend:
SimObject —
GFXSamplerStateData
```

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Structure</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GFXTextureArgument</td>
<td>resultArg</td>
<td>The selection of the destination register for the result of this stage. The default is GFXTACurrent.</td>
</tr>
<tr>
<td>GFXTextureTransformFlags</td>
<td>textureTransform</td>
<td>Sets the texture transform state. The default is GFXTTFDFDisable.</td>
</tr>
</tbody>
</table>

## Address Mode

<table>
<thead>
<tr>
<th>Structure</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GFXTextureAddressMode</td>
<td>addressModeU</td>
<td>The texture address mode for the u coordinate. The default is GFXAddressWrap.</td>
</tr>
<tr>
<td>GFXTextureAddressMode</td>
<td>addressModeV</td>
<td>The texture address mode for the v coordinate. The default is GFXAddressWrap.</td>
</tr>
<tr>
<td>GFXTextureAddressMode</td>
<td>addressModeW</td>
<td>The texture address mode for the w coordinate. The default is GFXAddressWrap.</td>
</tr>
</tbody>
</table>

## Alpha Op

<table>
<thead>
<tr>
<th>Structure</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GFXTextureArgument</td>
<td>alphaArg1</td>
<td>The first alpha argument for the texture stage. The default value is GFXTATexture.</td>
</tr>
<tr>
<td>GFXTextureArgument</td>
<td>alphaArg2</td>
<td>The second alpha argument for the texture stage. The default value is GFXTADiffuse.</td>
</tr>
</tbody>
</table>
GFXTextureArgument   alphaArg3
The third alpha channel selector operand for triadic operations (multiply, add, and linearly interpolate). The default value is GFXTACurrent.

GFXTextureOp    alphaOp
The texture alpha blending operation. The default value is GFXTOPModulate.

Color Op

GFXTextureArgument   colorArg1
The first color argument for the texture stage. The default value is GFXTACurrent.

GFXTextureArgument   colorArg2
The second color argument for the texture stage. The default value is GFXTATexture.

GFXTextureArgument   colorArg3
The third color argument for triadic operations (multiply, add, and linearly interpolate). The default value is GFXTACurrent.

GFXTextureOp    textureColorOp
The texture color blending operation. The default value is GFXTOPDisable which disables the sampler.

Filter State

GFXTextureFilterType   magFilter
The texture magnification filter. The
default is GFXTextureFilterLinear.

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><strong>maxAnisotropy</strong> The maximum texture anisotropy. The default value is 1.</td>
</tr>
<tr>
<td>GFXTextureFilterType</td>
<td><strong>minFilter</strong> The texture minification filter. The default is GFXTextureFilterLinear.</td>
</tr>
<tr>
<td>GFXTextureFilterType</td>
<td><strong>mipFilter</strong> The texture mipmap filter used during minification. The default is GFXTextureFilterLinear.</td>
</tr>
<tr>
<td>float</td>
<td><strong>mipLODBias</strong> The mipmap level of detail bias. The default value is zero.</td>
</tr>
</tbody>
</table>
Detailed Description

A sampler state used by GFXStateBlockData.

The samplers define how a texture will be sampled when used from the shader or fixed function device

Example:

```
singleton GFXSamplerStateData(SamplerClampLinear)
{
    textureColorOp = GFXTOPModulate;
    addressModeU = GFXAddressClamp;
    addressModeV = GFXAddressClamp;
    addressModeW = GFXAddressClamp;
    magFilter = GFXTextureFilterLinear;
    minFilter = GFXTextureFilterLinear;
    mipFilter = GFXTextureFilterLinear;
};
```

There are a few predefined samplers in the core scripts which you can use with GFXStateBlockData for the most common rendering cases:

- SamplerClampLinear
- SamplerClampPoint
- SamplerWrapLinear
- SamplerWrapPoint

See also:

GFXStateBlockData
Member Data Documentation

**GFXTextureAddressMode GFXSamplerStateData::addressModeU**

The texture address mode for the u coordinate. The default is GFXAddressWrap.

**GFXTextureAddressMode GFXSamplerStateData::addressModeV**

The texture address mode for the v coordinate. The default is GFXAddressWrap.

**GFXTextureAddressMode GFXSamplerStateData::addressModeW**

The texture address mode for the w coordinate. The default is GFXAddressWrap.

**GFXTextureArgument GFXSamplerStateData::alphaArg1**

The first alpha argument for the texture stage. The default value is GFXTATexture.

**GFXTextureArgument GFXSamplerStateData::alphaArg2**

The second alpha argument for the texture stage. The default value is GFXTADiffuse.

**GFXTextureArgument GFXSamplerStateData::alphaArg3**

The third alpha channel selector operand for triadic operations.
(multiply, add, and linearly interpolate). The default value is GFXTACurrent.

GFXTextureOp GFXSamplerStateData::alphaOp

The texture alpha blending operation. The default value is GFXTOPModulate.

GFXTextureArgument GFXSamplerStateData::colorArg1

The first color argument for the texture stage. The default value is GFXTACurrent.

GFXTextureArgument GFXSamplerStateData::colorArg2

The second color argument for the texture stage. The default value is GFXTATTexture.

GFXTextureArgument GFXSamplerStateData::colorArg3

The third color argument for triadic operations (multiply, add, and linearly interpolate). The default value is GFXTACurrent.

GFXTextureFilterType GFXSamplerStateData::magFilter

The texture magnification filter. The default is GFXTTextureFilterLinear.

int GFXSamplerStateData::maxAnisotropy
The maximum texture anisotropy. The default value is 1.

<table>
<thead>
<tr>
<th>GFXTextureFilterType GFXSamplerStateData::minFilter</th>
</tr>
</thead>
<tbody>
<tr>
<td>The texture minification filter. The default is GFXTextureFilterLinear.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GFXTextureFilterType GFXSamplerStateData::mipFilter</th>
</tr>
</thead>
<tbody>
<tr>
<td>The texture mipmap filter used during minification. The default is GFXTextureFilterLinear.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float GFXSamplerStateData::mipLODBias</th>
</tr>
</thead>
<tbody>
<tr>
<td>The mipmap level of detail bias. The default value is zero.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GFXTextureArgument GFXSamplerStateData::resultArg</th>
</tr>
</thead>
<tbody>
<tr>
<td>The selection of the destination register for the result of this stage. The default is GFXTACurrent.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GFXTextureOp GFXSamplerStateData::textureColorOp</th>
</tr>
</thead>
<tbody>
<tr>
<td>The texture color blending operation. The default value is GFXTOPDDisable which disables the sampler.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GFXTextureTransformFlags GFXSamplerStateData::textureTransform</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets the texture transform state. The default is GFXTTFFDisable.</td>
</tr>
</tbody>
</table>
GFXStateBlockData Class Reference

A state block description for rendering. More...

Inheritance diagram for GFXStateBlockData:

List of all members.
## Public Attributes

### Alpha Test

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>alphaDefined</td>
<td>Set to true if the alpha test state is not all defaults.</td>
</tr>
<tr>
<td>bool</td>
<td>alphaTestEnable</td>
<td>Enables per-pixel alpha testing. The default is false.</td>
</tr>
<tr>
<td>GFXCmpFunc</td>
<td>alphaTestFunc</td>
<td>The test function used to accept or reject a pixel based on its alpha value. The default is GFXCmpGreaterEqual.</td>
</tr>
<tr>
<td>int</td>
<td>alphaTestRef</td>
<td>The reference alpha value against which pixels are tested. The default is zero.</td>
</tr>
</tbody>
</table>

### Alpha Blending

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>blendDefined</td>
<td>Set to true if the alpha blend state is not all defaults.</td>
</tr>
<tr>
<td>GFXBlend</td>
<td>blendDest</td>
<td>The destination blend state. The default is GFXBlendZero.</td>
</tr>
<tr>
<td>bool</td>
<td>blendEnable</td>
<td>Enables alpha blending. The default is false.</td>
</tr>
<tr>
<td>GFXBlendOp</td>
<td>blendOp</td>
<td>The arithmetic operation applied to alpha blending. The default is GFXBlendOpAdd.</td>
</tr>
<tr>
<td>GFXBlend</td>
<td>blendSrc</td>
<td></td>
</tr>
</tbody>
</table>
The source blend state. The default is GFXBlendOne.

### Color Write

- **bool** `colorWriteAlpha`  
  Enables alpha channel writes. The default is true.

- **bool** `colorWriteBlue`  
  Enables blue channel writes. The default is true.

- **bool** `colorWriteDefined`  
  Set to true if the color write state is not all defaults.

- **bool** `colorWriteGreen`  
  Enables green channel writes. The default is true.

- **bool** `colorWriteRed`  
  Enables red channel writes. The default is true.

### Culling

- **bool** `cullDefined`  
  Set to true if the culling state is not all defaults.

- **GFXCullMode** `cullMode`  
  Defines how back facing triangles are culled if at all. The default is GFXCullCCW.

### Fixed Function

- **bool** `ffLighting`
Enables fixed function lighting when rendering without a shader on geometry with vertex normals. The default is false.

<table>
<thead>
<tr>
<th>bool</th>
<th>vertexColorEnable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Enables fixed function vertex coloring when rendering without a shader. The default is false.</td>
</tr>
</tbody>
</table>

### Sampler States

<table>
<thead>
<tr>
<th>bool</th>
<th>samplersDefined</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Set to true if the sampler states are not all defaults.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GFXSamplerStateData samplerStates [16]</th>
</tr>
</thead>
<tbody>
<tr>
<td>The array of texture sampler states.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ColorI</th>
<th>textureFactor</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The color used for multiple-texture blending with the GFXTATFactor texture-blending argument or the GFXTOPBlendFactorAlpha texture-blending operation. The default is opaque white (255, 255, 255, 255).</td>
</tr>
</tbody>
</table>

### Separate Alpha Blending

<table>
<thead>
<tr>
<th>bool</th>
<th>separateAlphaBlendDefined</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Set to true if the separate alpha blend state is not all defaults.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GFXBlend separateAlphaBlendDest</th>
</tr>
</thead>
<tbody>
<tr>
<td>The destination blend state. The default is GFXBlendZero.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>separateAlphaBlendEnable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Enables the separate blend mode for the alpha channel. The default is false.</td>
</tr>
</tbody>
</table>

| GFXBlendOp separateAlphaBlendOp |
The arithmetic operation applied to separate alpha blending. The default is GFXBlendOpAdd.

<table>
<thead>
<tr>
<th>GFXBlend</th>
<th>separateAlphaBlendSrc</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The source blend state. The default is GFXBlendOne.</td>
</tr>
</tbody>
</table>

### Stencil

<table>
<thead>
<tr>
<th>bool</th>
<th>stencilDefined</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Set to true if the stencil state is not all defaults.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>stencilEnable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Enables stenciling. The default is false.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GFXStencilOp</th>
<th>stencilFailOp</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The stencil operation to perform if the stencil test fails. The default is GFXStencilOpKeep.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GFXCmpFunc</th>
<th>stencilFunc</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The comparison function to test the reference value to a stencil buffer entry. The default is GFXCmpNever.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int</th>
<th>stencilMask</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The mask applied to the reference value and each stencil buffer entry to determine the significant bits for the stencil test. The default is 0xFFFFFFFF.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GFXStencilOp</th>
<th>stencilPassOp</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The stencil operation to perform if both the stencil and the depth tests pass. The default is GFXStencilOpKeep.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int</th>
<th>stencilRef</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The reference value for the stencil test. The default is zero.</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>----------</td>
<td>-----------------</td>
</tr>
<tr>
<td>int</td>
<td><code>stencilWriteMask</code></td>
</tr>
<tr>
<td>GFXStencilOp</td>
<td><code>stencilZFailOp</code></td>
</tr>
</tbody>
</table>

**Depth**

<table>
<thead>
<tr>
<th><strong>Type</strong></th>
<th><strong>Name</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>zBias</code></td>
<td>A floating-point bias used when comparing depth values. The default is zero.</td>
</tr>
<tr>
<td>bool</td>
<td><code>zDefined</code></td>
<td>Set to true if the depth state is not all defaults.</td>
</tr>
<tr>
<td>bool</td>
<td><code>zEnable</code></td>
<td>Enables z-buffer reads. The default is true.</td>
</tr>
<tr>
<td>GFXCmpFunc</td>
<td><code>zFunc</code></td>
<td>The depth comparison function which a pixel must pass to be written to the z-buffer. The default is GFXCmpLessEqual.</td>
</tr>
<tr>
<td>float</td>
<td><code>zSlopeBias</code></td>
<td>An additional floating-point bias based on the maximum depth slope of the triangle being rendered. The default is zero.</td>
</tr>
<tr>
<td>bool</td>
<td><code>zWriteEnable</code></td>
<td>Enables z-buffer writes. The default is true.</td>
</tr>
</tbody>
</table>
Detailed Description

A state block description for rendering.

This object is used with ShaderData in CustomMaterial and PostEffect to define the render state.

Example:

```cpp
singleton GFXStateBlockData( PFX_DOFDownSampleStateBlock )
{
    zDefined = true;
    zEnable = false;
    zWriteEnable = false;

    samplersDefined = true;
    samplerStates[0] = SamplerClampLinear;
    samplerStates[1] = SamplerClampPoint;

    // Copy the clamped linear sampler, but
    // the u coord to wrap for this special
    samplerStates[2] = new GFXSamplerStateData
    {
        addressModeU = GFXAddressWrap;
    };
};
```

Note:

The 'xxxxDefined' fields are used to know what groups of fields are modified when combining multiple state blocks in material processing. You should take care to enable the right ones when setting values.
Member Data Documentation

bool GFXStateBlockData::alphaDefined

Set to true if the alpha test state is not all defaults.

bool GFXStateBlockData::alphaTestEnable

Enables per-pixel alpha testing. The default is false.

GFXCmpFunc GFXStateBlockData::alphaTestFunc

The test function used to accept or reject a pixel based on its alpha value. The default is GFXCmpGreaterEqual.

int GFXStateBlockData::alphaTestRef

The reference alpha value against which pixels are tested. The default is zero.

bool GFXStateBlockData::blendDefined

Set to true if the alpha blend state is not all defaults.

GFXBlend GFXStateBlockData::blendDest

The destination blend state. The default is GFXBlendZero.

bool GFXStateBlockData::blendEnable
Enables alpha blending. The default is false.

<table>
<thead>
<tr>
<th>GFXBlendOp GFXStateBlockData::blendOp</th>
</tr>
</thead>
<tbody>
<tr>
<td>The arithmetic operation applied to alpha blending. The default is GFXBlendOpAdd.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GFXBlend GFXStateBlockData::blendSrc</th>
</tr>
</thead>
<tbody>
<tr>
<td>The source blend state. The default is GFXBlendOne.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool GFXStateBlockData::colorWriteAlpha</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enables alpha channel writes. The default is true.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool GFXStateBlockData::colorWriteBlue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enables blue channel writes. The default is true.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool GFXStateBlockData::colorWriteDefined</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set to true if the color write state is not all defaults.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool GFXStateBlockData::colorWriteGreen</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enables green channel writes. The default is true.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool GFXStateBlockData::colorWriteRed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>
Enables red channel writes. The default is true.

```cpp
bool GFXStateBlockData::cullDefined
```

Set to true if the culling state is not all defaults.

```cpp
GFXCullMode GFXStateBlockData::cullMode
```

Defines how back facing triangles are culled if at all. The default is GFXCullCCW.

```cpp
bool GFXStateBlockData::ffLighting
```

Enables fixed function lighting when rendering without a shader on geometry with vertex normals. The default is false.

```cpp
bool GFXStateBlockData::samplersDefined
```

Set to true if the sampler states are not all defaults.

```cpp
GFXSamplerStateData GFXStateBlockData::samplerStates[16]
```

The array of texture sampler states.

**Note:**

Not all graphics devices support 16 samplers. In general all systems support 4 samplers with most modern cards doing 8.

```cpp
bool GFXStateBlockData::separateAlphaBlendDefined
```
Set to true if the separate alpha blend state is not all defaults.

<table>
<thead>
<tr>
<th>GFXBlend GFXStateBlockData::separateAlphaBlendDest</th>
</tr>
</thead>
<tbody>
<tr>
<td>The destination blend state. The default is GFXBlendZero.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool GFXStateBlockData::separateAlphaBlendEnable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enables the separate blend mode for the alpha channel. The default is false.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GFXBlendOp GFXStateBlockData::separateAlphaBlendOp</th>
</tr>
</thead>
<tbody>
<tr>
<td>The arithmetic operation applied to separate alpha blending. The default is GFXBlendOpAdd.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GFXBlend GFXStateBlockData::separateAlphaBlendSrc</th>
</tr>
</thead>
<tbody>
<tr>
<td>The source blend state. The default is GFXBlendOne.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool GFXStateBlockData::stencilDefined</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set to true if the stencil state is not all defaults.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool GFXStateBlockData::stencilEnable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enables stenciling. The default is false.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GFXStencilOp GFXStateBlockData::stencilFailOp</th>
</tr>
</thead>
</table>
The stencil operation to perform if the stencil test fails. The default is GFXStencilOpKeep.

**GFXCmpFunc GFXStateBlockData::stencilFunc**

The comparison function to test the reference value to a stencil buffer entry. The default is GFXCmpNever.

**int GFXStateBlockData::stencilMask**

The mask applied to the reference value and each stencil buffer entry to determine the significant bits for the stencil test. The default is 0xFFFFFFFF.

**GFXStencilOp GFXStateBlockData::stencilPassOp**

The stencil operation to perform if both the stencil and the depth tests pass. The default is GFXStencilOpKeep.

**int GFXStateBlockData::stencilRef**

The reference value for the stencil test. The default is zero.

**int GFXStateBlockData::stencilWriteMask**

The write mask applied to values written into the stencil buffer. The default is 0xFFFFFFFF.

**GFXStencilOp GFXStateBlockData::stencilZFailOp**
The stencil operation to perform if the stencil test passes and the depth test fails. The default is GFXStencilOpKeep.

<table>
<thead>
<tr>
<th>Colorl GFXStateBlockData::textureFactor</th>
</tr>
</thead>
<tbody>
<tr>
<td>The color used for multiple-texture blending with the GFXTATFactor texture-blending argument or the GFXTOPBlendFactorAlpha texture-blending operation. The default is opaque white (255, 255, 255, 255).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool GFXStateBlockData::vertexColorEnable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enables fixed function vertex coloring when rendering without a shader. The default is false.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float GFXStateBlockData::zBias</th>
</tr>
</thead>
<tbody>
<tr>
<td>A floating-point bias used when comparing depth values. The default is zero.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool GFXStateBlockData::zDefined</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set to true if the depth state is not all defaults.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool GFXStateBlockData::zEnable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enables z-buffer reads. The default is true.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GFXCmpFunc GFXStateBlockData::zFunc</th>
</tr>
</thead>
</table>
The depth comparison function which a pixel must pass to be written to the z-buffer. The default is GFXCmpLessEqual.

```
float GFXStateBlockData::zSlopeBias
```

An additional floating-point bias based on the maximum depth slope of the triangle being rendered. The default is zero.

```
bool GFXStateBlockData::zWriteEnable
```

Enables z-buffer writes. The default is true.
GroundCover Class Reference

[Foliage]

Covers the ground in a field of objects (IE: Grass, Flowers, etc).

More...

Inheritance diagram for GroundCover:

```
SimObject

NetObject

SceneObject

GroundCover
```

List of all members.
## Public Attributes

### GroundCover General

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RectF</td>
<td>billboardUVs</td>
<td>[8] Subset material UV coordinates for this cover billboard.</td>
</tr>
<tr>
<td>float</td>
<td>clumpExponent</td>
<td>[8] An exponent used to bias between the minimum and maximum clump counts for a particular clump.</td>
</tr>
<tr>
<td>float</td>
<td>clumpRadius</td>
<td>[8] The maximum clump radius.</td>
</tr>
<tr>
<td>float</td>
<td>dissolveRadius</td>
<td>This is less than or equal to radius and defines when fading of cover elements begins.</td>
</tr>
<tr>
<td>int</td>
<td>gridSize</td>
<td>The number of cells per axis in the grid.</td>
</tr>
<tr>
<td>bool</td>
<td>invertLayer</td>
<td>[8] Indicates that the terrain material index given in 'layer' is an exclusion mask.</td>
</tr>
<tr>
<td>string</td>
<td>layer</td>
<td>[8] Terrain material name to limit coverage to, or blank to not limit.</td>
</tr>
<tr>
<td>string</td>
<td>Material</td>
<td>Material used by all GroundCover segments.</td>
</tr>
<tr>
<td>float</td>
<td>maxBillboardTiltAngle</td>
<td>The maximum amount of degrees the billboard will tilt down to match the camera.</td>
</tr>
<tr>
<td>int</td>
<td>maxClumpCount</td>
<td>[8] The maximum amount of elements in a clump.</td>
</tr>
<tr>
<td>int</td>
<td>maxElements</td>
<td>The maximum amount of cover elements to include in...</td>
</tr>
</tbody>
</table>
the grid at any one time.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float  maxElevation [8]</td>
<td>The maximum world space elevation for placement.</td>
</tr>
<tr>
<td>int    minClumpCount [8]</td>
<td>The minimum amount of elements in a clump.</td>
</tr>
<tr>
<td>float  minElevation [8]</td>
<td>The minimum world space elevation for placement.</td>
</tr>
<tr>
<td>float  probability [8]</td>
<td>The probability of one cover type verses another (relative to all cover types).</td>
</tr>
<tr>
<td>float  radius</td>
<td>Outer generation radius from the current camera position.</td>
</tr>
<tr>
<td>float  reflectScale</td>
<td>Scales the various culling radii when rendering a reflection. Typically for water.</td>
</tr>
<tr>
<td>int    seed</td>
<td>This RNG seed is saved and sent to clients for generating the same cover.</td>
</tr>
<tr>
<td>float  shapeCullRadius</td>
<td>This is the distance at which DTS elements are completely culled out.</td>
</tr>
<tr>
<td>filename shapeFilename [8]</td>
<td>The cover shape filename. [Optional].</td>
</tr>
<tr>
<td>bool   shapesCastShadows</td>
<td>Whether DTS elements should cast shadows or not.</td>
</tr>
<tr>
<td>float  sizeExponent [8]</td>
<td>An exponent used to bias between the minimum and maximum random sizes.</td>
</tr>
</tbody>
</table>
**float sizeMax [8]**  
The maximum random size of this cover type.

**float sizeMin [8]**  
The minimum random size for each cover type.

**float windScale [8]**  
The wind effect scale.

**float zOffset**  
Offset along the Z axis to render the ground cover.

---

**GroundCover Debug**

**bool lockFrustum**  
Debug parameter for locking the culling frustum which will freeze the cover generation.

**bool noBillboards**  
Debug parameter for turning off billboard rendering.

**bool noShapes**  
Debug parameter for turning off shape rendering.

**bool renderCells**  
Debug parameter for displaying the grid cells.

---

**GroundCover Wind**

**Point2F windDirection**  
The direction of the wind.

**float windGustFrequency**  
Controls how often the wind gust peaks per second.

**float windGustLength**  
The length in meters between peaks in the wind gust.

**float windGustStrength**  
The maximum distance in meters that the peak wind gust will displace an element.
<table>
<thead>
<tr>
<th>float</th>
<th>windTurbulenceFrequency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Controls the overall rapidity of the wind turbulence.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>windTurbulenceStrength</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The maximum distance in meters that the turbulence can displace a ground cover element.</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td><strong>isRenderable</strong></td>
<td>Enables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td><strong>isSelectable</strong></td>
<td>Enables selection of all instances of this type.</td>
</tr>
<tr>
<td>static int</td>
<td><strong>renderedBatches</strong></td>
<td>Stat for number of rendered billboard batches.</td>
</tr>
<tr>
<td>static int</td>
<td><strong>renderedBillboards</strong></td>
<td>Stat for number of rendered billboards.</td>
</tr>
<tr>
<td>static int</td>
<td><strong>renderedCells</strong></td>
<td>Stat for number of rendered cells.</td>
</tr>
<tr>
<td>static int</td>
<td><strong>renderedShapes</strong></td>
<td>Stat for number of rendered shapes.</td>
</tr>
</tbody>
</table>
Detailed Description

Covers the ground in a field of objects (IE: Grass, Flowers, etc).
**Member Data Documentation**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RectF GroundCover::billboardUVs[8]</td>
<td>Subset material UV coordinates for this cover billboard.</td>
</tr>
<tr>
<td>float GroundCover::clumpExponent[8]</td>
<td>An exponent used to bias between the minimum and maximum clump counts for a particular clump.</td>
</tr>
<tr>
<td>float GroundCover::clumpRadius[8]</td>
<td>The maximum clump radius.</td>
</tr>
<tr>
<td>float GroundCover::dissolveRadius</td>
<td>This is less than or equal to radius and defines when fading of cover elements begins.</td>
</tr>
<tr>
<td>int GroundCover::gridSize</td>
<td>The number of cells per axis in the grid.</td>
</tr>
<tr>
<td>bool GroundCover::invertLayer[8]</td>
<td>Indicates that the terrain material index given in 'layer' is an exclusion mask.</td>
</tr>
</tbody>
</table>
**string GroundCover::layer[8]**

Terrain material name to limit coverage to, or blank to not limit.

**bool GroundCover::lockFrustum**

Debug parameter for locking the culling frustum which will freeze the cover generation.

**string GroundCover::Material**

*Material* used by all *GroundCover* segments.

**float GroundCover::maxBillboardTiltAngle**

The maximum amount of degrees the billboard will tilt down to match the camera.

**int GroundCover::maxClumpCount[8]**

The maximum amount of elements in a clump.

**int GroundCover::maxElements**

The maximum amount of cover elements to include in the grid at any one time.

**float GroundCover::maxElevation[8]**

The maximum world space elevation for placement.
float GroundCover::maxSlope[8]

The maximum slope angle in degrees for placement.

int GroundCover::minClumpCount[8]

The minimum amount of elements in a clump.

float GroundCover::minElevation[8]

The minimum world space elevation for placement.

bool GroundCover::noBillboards

Debug parameter for turning off billboard rendering.

bool GroundCover::noShapes

Debug parameter for turning off shape rendering.

float GroundCover::probability[8]

The probability of one cover type verses another (relative to all cover types).

float GroundCover::radius

Outer generation radius from the current camera position.
<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>GroundCover::reflectScale</code></td>
<td>Scales the various culling radii when rendering a reflection. Typically for water.</td>
</tr>
<tr>
<td>bool</td>
<td><code>GroundCover::renderCells</code></td>
<td>Debug parameter for displaying the grid cells.</td>
</tr>
<tr>
<td>int</td>
<td><code>GroundCover::seed</code></td>
<td>This RNG seed is saved and sent to clients for generating the same cover.</td>
</tr>
<tr>
<td>float</td>
<td><code>GroundCover::shapeCullRadius</code></td>
<td>This is the distance at which DTS elements are completely culled out.</td>
</tr>
<tr>
<td>filename</td>
<td><code>GroundCover::shapeFilename</code></td>
<td>The cover shape filename. [Optional].</td>
</tr>
<tr>
<td>bool</td>
<td><code>GroundCover::shapesCastShadows</code></td>
<td>Whether DTS elements should cast shadows or not.</td>
</tr>
<tr>
<td>float</td>
<td><code>GroundCover::sizeExponent</code></td>
<td></td>
</tr>
</tbody>
</table>
An exponent used to bias between the minimum and maximum random sizes.

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>GroundCover::sizeMax[8]</code></td>
<td>The maximum random size of this cover type.</td>
</tr>
<tr>
<td>float</td>
<td><code>GroundCover::sizeMin[8]</code></td>
<td>The minimum random size for each cover type.</td>
</tr>
<tr>
<td>Point2F</td>
<td><code>GroundCover::windDirection</code></td>
<td>The direction of the wind.</td>
</tr>
<tr>
<td>float</td>
<td><code>GroundCover::windGustFrequency</code></td>
<td>Controls how often the wind gust peaks per second.</td>
</tr>
<tr>
<td>float</td>
<td><code>GroundCover::windGustLength</code></td>
<td>The length in meters between peaks in the wind gust.</td>
</tr>
<tr>
<td>float</td>
<td><code>GroundCover::windGustStrength</code></td>
<td>The maximum distance in meters that the peak wind gust will displace an element.</td>
</tr>
<tr>
<td>float</td>
<td><code>GroundCover::windScale[8]</code></td>
<td></td>
</tr>
</tbody>
</table>
The wind effect scale.

```plaintext
float GroundCover::windTurbulenceFrequency

Controls the overall rapidity of the wind turbulence.
```

```plaintext
float GroundCover::windTurbulenceStrength

The maximum distance in meters that the turbulence can displace a ground cover element.
```

```plaintext
float GroundCover::zOffset

Offset along the Z axis to render the ground cover.
```
GroundPlane Class Reference

[Terrain]

An infinite plane extending in all direction. More...

Inheritance diagram for GroundPlane:

```
  +-----------------+                     +-----------------+
  | SimObject       |                     | NetObject       |
  +-----------------+                     +-----------------+
                  |                     |                  |
                  |                      |                  |
  +-----------------+                     +-----------------+
  | SceneObject     |                     | GroundPlane     |
  +-----------------+                     +-----------------+ [legend]
```

List of all members.
Public Member Functions

void postApply ()

Intended as a helper to developers and editor scripts.
# Public Attributes

## Plane

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Material</td>
<td>string</td>
<td>Name of <strong>Material</strong> used to render GroundPlane's surface.</td>
</tr>
<tr>
<td>scaleU</td>
<td>float</td>
<td>Scale of texture repeat in the U direction.</td>
</tr>
<tr>
<td>scaleV</td>
<td>float</td>
<td>Scale of texture repeat in the V direction.</td>
</tr>
<tr>
<td>squareSize</td>
<td>float</td>
<td>Square size in meters to which GroundPlane subdivides its geometry.</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>static bool</th>
<th>isRenderable</th>
<th>Disables rendering of all instances of this type.</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
**Detailed Description**

An infinite plane extending in all direction.

GroundPlane is useful for setting up simple testing scenes, or it can be placed under an existing scene to keep objects from falling into 'nothing'.

GroundPlane may not be moved or rotated, it is always at the world origin.
Member Function Documentation

void GroundPlane::postApply()

Intended as a helper to developers and editor scripts.

Force trigger an inspectPostApply. This will transmit material and other fields to client objects.
Member Data Documentation

string GroundPlane::Material

Name of Material used to render GroundPlane's surface.

float GroundPlane::scaleU

Scale of texture repeat in the U direction.

float GroundPlane::scaleV

Scale of texture repeat in the V direction.

float GroundPlane::squareSize

Square size in meters to which GroundPlane subdivides its geometry.
GuiArrayCtrl Class Reference

Inheritance diagram for GuiArrayCtrl:

[legend]

Copyright © GarageGames, LLC. All Rights Reserved.
GuiAutoScrollCtrl Class Reference
[Container Controls]

A container that scrolls its child control up over time. More...

Inheritance diagram for GuiAutoScrollCtrl:

List of all members.
Public Member Functions

<table>
<thead>
<tr>
<th>void</th>
<th>reset ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reset scrolling.</td>
</tr>
</tbody>
</table>

Callbacks

<table>
<thead>
<tr>
<th>void</th>
<th>onComplete ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Called when the child control has been scrolled in entirety.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>onReset ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Called when the child control is reset to its initial position and the cycle starts again.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>onStart ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Called when the control starts to scroll.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>onTick ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Called every 32ms on the control.</td>
</tr>
</tbody>
</table>
## Public Attributes

### Scrolling

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>childBorder</td>
<td>Padding to put around child control (in pixels).</td>
</tr>
<tr>
<td>bool</td>
<td>isLooping</td>
<td>If true, the scrolling will reset to the beginning once completing a cycle.</td>
</tr>
<tr>
<td>float</td>
<td>resetDelay</td>
<td>Seconds to wait after scrolling completes before resetting and starting over.</td>
</tr>
<tr>
<td>GuiAutoScrollDirection</td>
<td>scrollDirection</td>
<td>Direction in which the child control is moved.</td>
</tr>
<tr>
<td>bool</td>
<td>scrollOutOfSight</td>
<td>If true, the child control will be completely scrolled out of sight; otherwise it will only scroll until the other end becomes visible.</td>
</tr>
<tr>
<td>float</td>
<td>scrollSpeed</td>
<td>Scrolling speed in pixels per second.</td>
</tr>
<tr>
<td>float</td>
<td>startDelay</td>
<td>Seconds to wait before starting to scroll.</td>
</tr>
</tbody>
</table>
Detailed Description

A container that scrolls its child control up over time.

This container can be used to scroll a single child control in either of the four directions.

Example:

```cpp
// Create a GuiAutoScrollCtrl that scrolls
new GuiAutoScrollCtrl( CreditsScroller )
{
    position = "0 0";
    extent = Canvas.extent.x SPC Canvas.extent.y;

    scrollDirection = "Up"; // Scroll upwards.
    startDelay = 4; // Wait 4 seconds before
    isLooping = false; // Don't loop the credits.
    scrollOutOfSight = true; // Scroll up for

    new GuiMLTextCtrl()
    {
        text = $CREDITS;
    };
};

function CreditsScroller::onComplete( %this
{
    // Switch back to main menu after credits
    Canvas.setContent( MainMenu );
}

// Start rolling credits.
Canvas.setContent( CreditsScroller );
```
Note:

Only the first child will be scrolled.
Member Function Documentation

void GuiAutoScrollCtrl::onComplete()

Called when the child control has been scrolled in entirety.

void GuiAutoScrollCtrl::onReset()

Called when the child control is reset to its initial position and the cycle starts again.

void GuiAutoScrollCtrl::onStart()

Called when the control starts to scroll.

void GuiAutoScrollCtrl::onTick()

Called every 32ms on the control.

void GuiAutoScrollCtrl::reset()

Reset scrolling.
**Member Data Documentation**

**int GuiAutoScrollCtrl::childBorder**

Padding to put around child control (in pixels).

**bool GuiAutoScrollCtrl::isLooping**

If true, the scrolling will reset to the beginning once completing a cycle.

**float GuiAutoScrollCtrl::resetDelay**

Seconds to wait after scrolling completes before resetting and starting over.

**Note:**
Only takes effect if *isLooping* is true.

**GuiAutoScrollDirection GuiAutoScrollCtrl::scrollDirection**

Direction in which the child control is moved.

**bool GuiAutoScrollCtrl::scrollOutOfSight**

If true, the child control will be completely scrolled out of sight; otherwise it will only scroll until the other end becomes visible.

**float GuiAutoScrollCtrl::scrollSpeed**
Scrolling speed in pixels per second.

```
float GuiAutoScrollCtrl::startDelay
```

Seconds to wait before starting to scroll.
GuiBitmapBorderCtrl Class Reference
[Image and Video Controls]

A control that renders a skinned border specified in its profile. More...

Inheritance diagram for GuiBitmapBorderCtrl:

```
SimObject

| SimSet |

| SimGroup |

| GuiControl |

GuiBitmapBorderCtrl
```

List of all members.
**Detailed Description**

A control that renders a skinned border specified in its profile.

This control uses the bitmap specified in its profile (GuiControlProfile::bitmapName). It takes this image and breaks up aspects of it to skin the border of this control with. It is also important to set `GuiControlProfile::hasBitmapArray` to true on the profile as well.

The bitmap referenced should be broken up into a 3 x 3 grid (using the top left color pixel as a border color between each of the images) in which it will map to the following places: 1 = Top Left Corner 2 = Top Right Corner 3 = Top Center 4 = Left Center 5 = Right Center 6 = Bottom Left Corner 7 = Bottom Center 8 = Bottom Right Corner 0 = Nothing

1 2 3 4 5 0 6 7 8

**Example:**

```cpp
singleton GuiControlProfile (BorderGUIProfile) {
    bitmap = "core/art/gui/images/borderArray"
    hasBitmapArray = true;
    opaque = false;
};

new GuiBitmapBorderCtrl(BitmapBorderGUI) {
    profile = "BorderGUIProfile";
    position = "0 0";
    extent = "400 40";
    visible = "1";
};
```

**See also:**
GuiControlProfile::bitmapName
GuiControlProfile::hasBitmapArray

Copyright © GarageGames, LLC. All Rights Reserved.
GuiBitmapButtonCtrl Class Reference
[Button Controls]

A button that renders its various states (mouse over, pushed, etc.) from separate bitmaps. More...

Inheritance diagram for GuiBitmapButtonCtrl:

List of all members.
Public Member Functions

void setBitmap (string path)
    Set the bitmap to show on the button.

Callbacks

void onAltClick ()
    Called when per-modifier functionality is enabled and the user clicks on the button with the ALT key pressed.

void onCtrlClick ()
    Called when per-modifier functionality is enabled and the user clicks on the button with the CTRL key pressed.

void onDefaultClick ()
    Called when per-modifier functionality is enabled and the user clicks on the button without any modifier pressed.

void onShiftClick ()
    Called when per-modifier functionality is enabled and the user clicks on the button with the SHIFT key pressed.
## Public Attributes

### Bitmap

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><strong>autoFitExtents</strong> If true, the control's extents will be set to match the bitmap's extents when setting the bitmap.</td>
</tr>
<tr>
<td>filename</td>
<td><strong>bitmap</strong> Texture file to display on this button.</td>
</tr>
<tr>
<td>GuiBitmapMode</td>
<td><strong>bitmapMode</strong> Behavior for fitting the bitmap to the control extents.</td>
</tr>
<tr>
<td>bool</td>
<td><strong>useModifiers</strong> If true, per-modifier button functionality is enabled.</td>
</tr>
<tr>
<td>bool</td>
<td><strong>useStates</strong> If true, per-mouse state button functionality is enabled.</td>
</tr>
</tbody>
</table>
Detailed Description

A button that renders its various states (mouse over, pushed, etc.) from separate bitmaps.

A bitmapped button is a push button that uses one or more texture images for rendering its individual states.

To find the individual textures associated with the button, a naming scheme is used. For each state a suffix is appended to the texture file name given in the GuiBitmapButtonCtrl::bitmap field:

- "_n": Normal state. This one will be active when no other state applies.
- "_h": Highlighted state. This applies when the mouse is hovering over the button.
- "_d": Depressed state. This applies when the left mouse button has been clicked on the button but not yet released.
- "_i": Inactive state. This applies when the button control has been deactivated (GuiControl::setActive())

If a bitmap for a particular state cannot be found, the default bitmap will be used. To disable all state-based bitmap functionality, set useStates to false which will make the control solely render from the bitmap specified in the bitmap field.
Per-Modifier Button Actions

If GuiBitmapButtonCtrl::useModifiers is set to true, per-modifier button actions and textures are enabled. This functionality allows to associate different images and different actions with a button depending on which modifiers are pressed on the keyboard by the user.

When enabled, this functionality alters the texture lookup above by prepending the following strings to the suffixes listed above:

- "": Default. No modifier is pressed.
- "_ctrl": Image to use when CTRL/CMD is down.
- "_alt": Image to use when ALT is down.
- "_shift": Image to use when SHIFT is down

When this functionality is enabled, a new set of callbacks is used:

- onDefaultClick: Button was clicked without a modifier being pressed.
- onCtrlClick: Button was clicked with the CTRL/CMD key down.
- onAltClick: Button was clicked with the ALT key down.
- onShiftClick: Button was clicked with the SHIFT key down.

GuiControl::command or GuiControl::onAction() still work as before when per-modifier functionality is enabled.

Note that modifiers cannot be mixed. If two or more modifiers are pressed, a single one will take precedence over the remaining modifiers. The order of precedence corresponds to the order listed above.

Example:

```c++
// Create an OK button that will trigger an onOk() call on its parent when clicked:
%okButton = new GuiBitmapButtonCtrl()
{
    bitmap = "art/gui/okButton";
    autoFitExtents = true;
}
```
command = "$ThisControl.getParent().onOk();"
Member Function Documentation

void GuiBitmapButtonCtrl::onAltClick()

Called when per-modifier functionality is enabled and the user clicks on the button with the ALT key pressed.

Per-Modifier Button Actions

void GuiBitmapButtonCtrl::onCtrlClick()

Called when per-modifier functionality is enabled and the user clicks on the button with the CTRL key pressed.

Per-Modifier Button Actions

void GuiBitmapButtonCtrl::onDefaultClick()

Called when per-modifier functionality is enabled and the user clicks on the button without any modifier pressed.

Per-Modifier Button Actions

void GuiBitmapButtonCtrl::onShiftClick()

Called when per-modifier functionality is enabled and the user clicks on the button with the SHIFT key pressed.

Per-Modifier Button Actions

void GuiBitmapButtonCtrl::setBitmap(string path)
Set the bitmap to show on the button.

**Parameters:**

`path Path` to the texture file in any of the supported formats.
**Member Data Documentation**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| `bool GuiBitmapButtonCtrl::autoFitExtents` | If true, the control's extents will be set to match the bitmap's extents when setting the bitmap.  
The bitmap extents will always be taken from the default/normal bitmap (in case the extents of the various bitmaps do not match up.) |
| `filename GuiBitmapButtonCtrl::bitmap` | Texture file to display on this button.  
If useStates is false, this will be the file that renders on the control.  
Otherwise, this will specify the default texture name to which the various state and modifier suffixes are appended to find the per-state and per-modifier (if enabled) textures. |
| `GuiBitmapMode GuiBitmapButtonCtrl::bitmapMode` | Behavior for fitting the bitmap to the control extents.  
If set to 'Stretched', the bitmap will be stretched both vertically and horizontally to fit inside the control's extents.  
If set to 'Centered', the bitmap will stay at its original resolution centered in the control's rectangle (getting clipped if the control is smaller than the texture). |
| `bool GuiBitmapButtonCtrl::useModifiers` | If true, per-modifier button functionality is enabled. |
Per-Modifier Button Actions

bool GuiBitmapButtonCtrl::useStates

If true, per-mouse state button functionality is enabled.

Defaults to true.

If you do not use per-state images on this button set this to false to speed up the loading process by inhibiting searches for the individual images.
GuiBitmapButtonTextCtrl Class Reference

[Button Controls]

An extension of GuiBitmapButtonCtrl that additionally renders a text label on the bitmapped button. More...

Inheritance diagram for GuiBitmapButtonTextCtrl:

```
SimObject
  ↓
SimSet
  ↓
SimGroup
  ↓
GuiControl
  ↓
GuiButtonBaseCtrl
  ↓
GuiButtonCtrl
  ↓
GuiBitmapButtonCtrl
  ↓
GuiBitmapButtonTextCtrl
```

List of all members.
**Detailed Description**

An extension of `GuiBitmapButtonCtrl` that additionally renders a text label on the bitmapped button.

The text for the label is taken from the `GuiButtonBaseCtrl::text` property.

For rendering, the label is placed, relative to the control's upper left corner, at the text offset specified in the control's profile `(GuiControlProfile::textOffset)` and justified according to the profile's setting `(GuiControlProfile::justify)`.

**See also:**

- `GuiControlProfile::textOffset`
- `GuiControlProfile::justify`

Copyright © GarageGames, LLC. All Rights Reserved.
GuiBitmapCtrl Class Reference
[General Controls]

A gui control that is used to display an image. More...

Inheritance diagram for GuiBitmapCtrl:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void <code>setBitmap</code> (String filename, bool resize)</td>
<td>Assign an image to the control.</td>
</tr>
<tr>
<td>void <code>setBitmap</code> (String filename)</td>
<td>Assign an image to the control.</td>
</tr>
<tr>
<td>void <code>setValue</code> (int x, int y)</td>
<td>Set the offset of the bitmap within the control.</td>
</tr>
</tbody>
</table>
## Public Attributes

### Bitmap

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filename</td>
<td><code>bitmap</code></td>
</tr>
<tr>
<td>bool</td>
<td><code>wrap</code></td>
</tr>
</tbody>
</table>
Detailed Description

A gui control that is used to display an image.

The image is stretched to the constraints of the control by default. However, the control can also tile the image as well.

The image itself is stored inside the `GuiBitmapCtrl::bitmap` field. The boolean value that decides whether the image is stretched or tiled is stored inside the `GuiBitmapCtrl::wrap` field.

Example:

```c++
// Create a tiling GuiBitmapCtrl that displays "myImage.png"
GuiBitmapCtrl* bitmapCtrl = new GuiBitmapCtrl()
{
    bitmap = "myImage.png";
    wrap = "true";
};
```
**Member Function Documentation**

```cpp
void GuiBitmapCtrl::setBitmap(String filename, bool resize)
```

Assign an image to the control.

Child controls with resize according to their layout settings.

**Parameters:**

- `filename` The filename of the image.
- `resize` Optional parameter. If true, the GUI will resize to fit the image.

```cpp
void GuiBitmapCtrl::setBitmap(String filename)
```

Assign an image to the control.

Child controls will resize according to their layout settings.

**Parameters:**

- `filename` The filename of the image.
- `resize` A boolean value that decides whether the ctrl refreshes or not.

```cpp
void GuiBitmapCtrl::setValue(int x, int y)
```

Set the offset of the bitmap within the control.

**Parameters:**

- `x` The x-axis offset of the image.
$y$ The $y$-axis offset of the image.
Member Data Documentation

<table>
<thead>
<tr>
<th>filename GuiBitmapCtrl::bitmap</th>
</tr>
</thead>
</table>

The bitmap file to display in the control.

<table>
<thead>
<tr>
<th>bool GuiBitmapCtrl::wrap</th>
</tr>
</thead>
</table>

If true, the bitmap is tiled inside the control rather than stretched to fit.

Copyright © GarageGames, LLC. All Rights Reserved.
GuiBorderButtonCtrl Class Reference
[Button Controls]

A push button that renders only a border. More...

Inheritance diagram for GuiBorderButtonCtrl:

```
SimObject
  ^
  |  
SimSet
  ^
  |  
SimGroup
  ^
  |  
GuiControl
  ^
  |  
GuiButtonBaseCtrl
  ^
  |  
GuiBorderButtonCtrl
```

List of all members.
Detailed Description

A push button that renders only a border.

A border button consists of a border rendered along its extents according to the border thickness defined in its profile (GuiControlProfile::border). For the border color, a color is selected from the profile according to current button state:

- Default state: GuiControlProfile::borderColor
- Highlighted (mouse is over the button): GuiControlProfile::fontColorHL
- Depressed (mouse button down but not yet released): GuiControlProfile::fontColorSEL
GuiBubbleTextCtrl Class Reference
[General Controls]

A single-line text control that displays its text in a multi-line popup when clicked. More...

Inheritance diagram for GuiBubbleTextCtrl:

```
SimObject
    ▼
    SimSet
        ▼
        SimGroup
            ▼
            GuiControl
                ▼
                GuiContainer
                    ▼
                    GuiTextCtrl
                        ▼
                        GuiBubbleTextCtrl
```

Legend

List of all members.
**Detailed Description**

A single-line text control that displays its text in a multi-line popup when clicked.

This control acts like a `GuiTextCtrl` (and inherits from it), when clicked it creates a `GuiMLTextCtrl` roughly where you clicked with the same text in it. This allows you to have a single line text control which upon clicking will display the entire text contained in a multi-line format.

**Example:**

```java
new GuiBubbleTextCtrl(BubbleTextGUI) {
    text = "This is the first sentence. This second sentence can be sized outside of the default single line view, upon clicking this will be displayed in a multi-line format.";
};
```

**See also:**

- `GuiTextCtrl`
- `GuiMLTextCtrl`
GuiButtonBaseCtrl Class Reference
[Button Controls]

The base class for the various button controls. More...

Inheritance diagram for GuiButtonBaseCtrl:

- SimObject
- SimSet
- SimGroup
- GuiControl
- GuiButtonBaseCtrl
- GuiBorderButtonCtrl
- GuiButtonCtrl
- GuiCheckBoxCtrl
- GuiSwitchButtonCtrl
- GuiMapButtonCtrl
- GuiIconButtonCtrl
- GuiToggleButtonCtrl
- GuiRadioButtonCtrl
- GuiBitmapButtonTextCtrl

[legend]

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>string getText ()</code></td>
<td>Get the text display on the button's label (if any).</td>
</tr>
<tr>
<td><code>void performClick ()</code></td>
<td>Simulate a click on the button.</td>
</tr>
<tr>
<td><code>void resetState ()</code></td>
<td>Reset the mousing state of the button.</td>
</tr>
<tr>
<td><code>void setStateOn (bool isOn=true)</code></td>
<td>For toggle or radio buttons, set whether the button is currently activated or not. For radio buttons, toggling a button on will toggle all other radio buttons in its group to off.</td>
</tr>
<tr>
<td><code>void setText (string text)</code></td>
<td>Set the text displayed on the button's label.</td>
</tr>
<tr>
<td><code>void setTextID (string id)</code></td>
<td>Set the text displayed on the button's label using a string from the string table assigned to the control.</td>
</tr>
</tbody>
</table>

### Callbacks

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void onClick ()</code></td>
<td>Called when the primary action of the button is triggered (e.g. by a left mouse click).</td>
</tr>
<tr>
<td><code>void onDoubleClick ()</code></td>
<td>Called when the left mouse button is double-clicked on the button.</td>
</tr>
<tr>
<td><code>void onMouseDown ()</code></td>
<td>If <code>useMouseEvents</code> is true, this is called when the left mouse button is pressed on an (active) button.</td>
</tr>
<tr>
<td><code>void onMouseDragged ()</code></td>
<td>If <code>useMouseEvents</code> is true, this is called when a left mouse button drag is detected, i.e. when the user pressed the left button.</td>
</tr>
</tbody>
</table>
mouse button on the control and then moves the mouse over a certain distance threshold with the mouse button still pressed.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void onMouseEnter()</td>
<td>If <code>useMouseEvents</code> is true, this is called when the mouse cursor moves over the button (only if the button is the front-most visible control, though).</td>
</tr>
<tr>
<td>void onMouseLeave()</td>
<td>If <code>useMouseEvents</code> is true, this is called when the mouse cursor moves off the button (only if the button had previously received an <code>onMouseEvent()</code> event).</td>
</tr>
<tr>
<td>void onMouseUp()</td>
<td>If <code>useMouseEvents</code> is true, this is called when the left mouse button is release over an (active) button.</td>
</tr>
<tr>
<td>void onClick()</td>
<td>Called when the right mouse button is clicked on the button.</td>
</tr>
</tbody>
</table>
## Public Attributes

### Button

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GuiButtonType</td>
<td><code>buttonType</code></td>
</tr>
<tr>
<td>int</td>
<td><code>groupNum</code></td>
</tr>
<tr>
<td>caseString</td>
<td><code>text</code></td>
</tr>
<tr>
<td>string</td>
<td><code>textID</code></td>
</tr>
<tr>
<td>bool</td>
<td><code>useMouseEvents</code></td>
</tr>
</tbody>
</table>
Detailed Description

The base class for the various button controls.

This is the base class for the various types of button controls. If no more specific functionality is required than offered by this class, then it can be instantiated and used directly. Otherwise, its subclasses should be used:

- **GuiRadioCtrl** (radio buttons)
- **GuiCheckBoxCtrl** (checkboxes)
- **GuiButtonCtrl** (push buttons with text labels)
- **GuiBitmapButtonCtrl** (bitmapped buttons)
- **GuiBitmapButtonTextCtrl** (bitmapped buttons with a text label)
- **GuiToggleButtonCtrl** (toggle buttons, i.e. push buttons with "sticky" behavior)
- **GuiSwatchButtonCtrl** (color swatch buttons)
- **GuiBorderButtonCtrl** (push buttons for surrounding child controls)
Member Function Documentation

string GuiButtonBaseCtrl::getText( )

Get the text display on the button's label (if any).

**Returns:**

The button's label.

void GuiButtonBaseCtrl::onClick( )

Called when the primary action of the button is triggered (e.g. by a left mouse click).

void GuiButtonBaseCtrl::onDoubleClick( )

Called when the left mouse button is double-clicked on the button.

void GuiButtonBaseCtrl::onMouseDown( )

If useMouseEvents is true, this is called when the left mouse button is pressed on an (active) button.

void GuiButtonBaseCtrl::onMouseDragged( )

If useMouseEvents is true, this is called when a left mouse button drag is detected, i.e. when the user pressed the left mouse button on the control and then moves the mouse over a certain distance threshold with the mouse button still pressed.
void GuiButtonBaseCtrl::onMouseEnter()

If `useMouseEvents` is true, this is called when the mouse cursor moves over the button (only if the button is the front-most visible control, though).

void GuiButtonBaseCtrl::onMouseLeave()

If `useMouseEvents` is true, this is called when the mouse cursor moves off the button (only if the button had previously received an `onMouseEvent()` event).

void GuiButtonBaseCtrl::onMouseUp()

If `useMouseEvents` is true, this is called when the left mouse button is release over an (active) button.

**Note:**
To trigger actions, better use `onClick()` since `onMouseUp()` will also be called when the mouse was not originally pressed on the button.

void GuiButtonBaseCtrl::onRightClick()

Called when the right mouse button is clicked on the button.

void GuiButtonBaseCtrl::performClick()

Simulate a click on the button.

This method will trigger the button's action just as if the button had been pressed by the user.
void GuiButtonBaseCtrl::resetState(
)

Reset the mousing state of the button.
This method should not generally be called.

void GuiButtonBaseCtrl::setStateOn(bool isOn = true)

For toggle or radio buttons, set whether the button is currently activated or not. For radio buttons, toggling a button on will toggle all other radio buttons in its group to off.

**Parameters:**

- `isOn` If true, the button will be toggled on (if not already); if false, it will be toggled off.

**Note:**

Toggling the state of a button with this method will not trigger the action associated with the button. To do that, use `performClick()`.

Reimplemented in `GuiCheckBoxCtrl`.

void GuiButtonBaseCtrl::setText(string text)

Set the text displayed on the button's label.

**Parameters:**

- `text` The text to display as the button's text label.

**Note:**

Not all buttons render text labels.

**See also:**
void GuiButtonBaseCtrl::setTextID (string id)

Set the text displayed on the button's label using a string from the string table assigned to the control.

Parameters:

- id  Name of the variable that contains the integer string ID. Used to look up string in table.

Note:

Not all buttons render text labels.

See also:

- setText
- getText
- GuiControl::langTableMod
- LangTable

Internationalization
### Member Data Documentation

<table>
<thead>
<tr>
<th><strong>GuiButtonType GuiButtonBaseCtrl::buttonType</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Button behavior type.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>int GuiButtonBaseCtrl::groupNum</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Radio button toggle group number. All radio buttons that are assigned the same groupNum and that are parented to the same control will synchronize their toggle state, i.e. if one radio button is toggled on all other radio buttons in its group will be toggled off. The default group is -1.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>caseString GuiButtonBaseCtrl::text</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Text label to display on button (if button class supports text labels).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>string GuiButtonBaseCtrl::textID</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>ID of string in string table to use for text label on button.</td>
</tr>
</tbody>
</table>

**See also:**
- setTextID
- GuiControl::langTableMod
- LangTable

<table>
<thead>
<tr>
<th><strong>bool GuiButtonBaseCtrl::useMouseEvents</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>If true, mouse events will be passed on to script. Default is false.</td>
</tr>
</tbody>
</table>
GuiButtonCtrl Class Reference

[Button Controls]

The most widely used button class. More...

Inheritance diagram for GuiButtonCtrl:

List of all members.
Detailed Description

The most widely used button class.

GuiButtonCtrl renders separately of, but utilizes all of the functionality of GuiBaseButtonCtrl. This grants GuiButtonCtrl the versatility to be either of the 3 button types.

Example:

```java
// Create a PushButton GuiButtonCtrl that calls randomFunction when clicked
%button = new GuiButtonCtrl()
{
    profile = "GuiButtonProfile";
    buttonType = "PushButton";
    command = "randomFunction();";
};
```
GuiCanvas Class Reference
[Core Controls]

A canvas on which rendering occurs. More...

Inheritance diagram for GuiCanvas:

```
    SimObject
      |
      v
    SimSet
      |
      v
    SimGroup
      |
      v
  GuiControl
      |
      v
GuiCanvas

[legend]
```

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Point2I</th>
<th><code>clientToScreen</code> (Point2I coordinate)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Translate a coordinate from canvas window-space to screen-space.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th><code>cursorOff</code> ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Turns on the mouse off.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th><code>cursorOn</code> ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Turns on the mouse cursor.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int</th>
<th><code>getContent</code> ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Get the <code>GuiControl</code> which is being used as the content.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Point2I</th>
<th><code>getCursorPos</code> ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Get the current position of the cursor.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Point2I</th>
<th><code>getExtent</code> ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Returns the dimensions of the canvas.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th><code>getMode</code> (int modeId)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Gets information on the specified mode of this device.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int</th>
<th><code>getModeCount</code> ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Gets the number of modes available on this device.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int</th>
<th><code>getMouseControl</code> ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Gets the gui control under the mouse.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th><code>getVideoMode</code> ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Gets the current screen mode as a string.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Point2I</th>
<th><code>getWindowPosition</code> ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Get the current position of the platform window associated with the canvas.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th><code>hideCursor</code> ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disable rendering of the cursor.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th><code>isCursorOn</code> ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Determines if mouse cursor is enabled.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>bool isCursorShown ()</td>
<td>Determines if mouse cursor is rendering.</td>
</tr>
<tr>
<td>bool isFullscreen ()</td>
<td>Is this canvas currently fullscreen?</td>
</tr>
<tr>
<td>bool isMaximized ()</td>
<td></td>
</tr>
<tr>
<td>bool isMinimized ()</td>
<td></td>
</tr>
<tr>
<td>void maximizeWindow ()</td>
<td>maximize this canvas' window.</td>
</tr>
<tr>
<td>void minimizeWindow ()</td>
<td>minimize this canvas' window.</td>
</tr>
<tr>
<td>void popDialog (GuiControl ctrl)</td>
<td>Removes a specific dialog control.</td>
</tr>
<tr>
<td>void popDialog ()</td>
<td>Removes a dialog at the front most layer.</td>
</tr>
<tr>
<td>void popLayer ()</td>
<td>Removes the top most layer of dialogs.</td>
</tr>
<tr>
<td>void popLayer (S32 layer)</td>
<td>Removes a specified layer of dialogs.</td>
</tr>
<tr>
<td>void pushDialog (GuiControl ctrl, int layer=0, bool center=false)</td>
<td>Adds a dialog control onto the stack of dialogs.</td>
</tr>
<tr>
<td>void renderFront (bool enable)</td>
<td>This turns on/off front-buffer rendering.</td>
</tr>
<tr>
<td>void repaint (int elapsedMS=0)</td>
<td>Force canvas to redraw. If the elapsed time is greater than the time since the last paint then the repaint will be skipped.</td>
</tr>
<tr>
<td>void reset ()</td>
<td>Reset the update regions for the canvas.</td>
</tr>
<tr>
<td>void restoreWindow ()</td>
<td>restore this canvas' window.</td>
</tr>
<tr>
<td>Point2I screenToClient (Point2I coordinate)</td>
<td></td>
</tr>
</tbody>
</table>
Translate a coordinate from screen-space to canvas window-space.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void setContent (GuiControl ctrl)</td>
<td>Set the content of the canvas to a specified control.</td>
</tr>
<tr>
<td>void setCursor (GuiCursor cursor)</td>
<td>Sets the cursor for the canvas.</td>
</tr>
<tr>
<td>bool setCursorPos (Point2I pos)</td>
<td>Sets the position of the cursor.</td>
</tr>
<tr>
<td>bool setCursorPos (F32 posX, F32 posY)</td>
<td>Sets the position of the cursor.</td>
</tr>
<tr>
<td>void setFocus ()</td>
<td>Claim OS input focus for this canvas' window.</td>
</tr>
<tr>
<td>void setVideoMode (int width, int height, bool fullscreen,[int bitDepth],[int refreshRate],[int antialiasLevel])</td>
<td>Change the video mode of this canvas. This method has the side effect of setting the $pref::Video::mode to the new values.</td>
</tr>
<tr>
<td>void setWindowPosition (Point2I position)</td>
<td>Set the position of the platform window associated with the canvas.</td>
</tr>
<tr>
<td>void setWindowTitle (string newTitle)</td>
<td>Change the title of the OS window.</td>
</tr>
<tr>
<td>void showCursor ()</td>
<td>Enable rendering of the cursor.</td>
</tr>
<tr>
<td>void toggleFullscreen ()</td>
<td>toggle canvas from fullscreen to windowed mode or back.</td>
</tr>
</tbody>
</table>
### Public Attributes

#### Mouse Handling

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>alwaysHandleMouseButtons</code></td>
<td>Deal with mouse buttons, even if the cursor is hidden.</td>
</tr>
</tbody>
</table>

#### Canvas Rendering

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><code>numFences</code></td>
<td>The number of GFX fences to use.</td>
</tr>
</tbody>
</table>
Detailed Description

A canvas on which rendering occurs.
What a GUICanvas Can Contain...

Content Control

A content control is the top level GuiControl for a screen. This GuiControl will be the parent control for all other GuiControls on that particular screen.

Dialogs

A dialog is essentially another screen, only it gets overlaid on top of the current content control, and all input goes to the dialog. This is most akin to the "Open File" dialog box found in most operating systems. When you choose to open a file, and the "Open File" dialog pops up, you can no longer send input to the application, and must complete or cancel the open file request. Torque keeps track of layers of dialogs. The dialog with the highest layer is on top and will get all the input, unless the dialog is modeless, which is a profile option.

See also:

GuiControlProfile
Dirty Rectangles

The **GuiCanvas** is based on dirty regions. Every frame the canvas paints only the areas of the canvas that are 'dirty' or need updating. In most cases, this only is the area under the mouse cursor. This is why if you look in guiCanvas.cc the call to glClear is commented out. What you will see is a black screen, except in the dirty regions, where the screen will be painted normally. If you are making an animated **GuiControl** you need to add your control to the dirty areas of the canvas.

**See also:**

**GuiControl**
Member Function Documentation

Point2I GuiCanvas::clientToScreen(Point2I coordinate)

Translate a coordinate from canvas window-space to screen-space.

Parameters:

coordinate The coordinate in window-space.

Returns:

The given coordinate translated to screen-space.

void GuiCanvas::cursorOff()

Turns on the mouse off.

Example:

Canvas.cursorOff();

void GuiCanvas::cursorOn()

Turns on the mouse cursor.

Example:

Canvas.cursorOn();

int GuiCanvas::getContent()

Get the GuiControl which is being used as the content.
Example:

```c++
Canvas.getContent();
```

**Returns:**
ID of current content control

```c++
Point2I GuiCanvas::getCursorPos()
```

Get the current position of the cursor.

**Parameters:**

`param` Description

Example:

```c++
%cursorPos = Canvas.getCursorPos();
```

**Returns:**
Screen coordinates of mouse cursor, in format "X Y"

```c++
Point2I GuiCanvas::getExtent()
```

Returns the dimensions of the canvas.

Example:

```c++
%extent = Canvas.getExtent();
```

**Returns:**
Width and height of canvas. Formatted as numerical values in a single string "# #"

Reimplemented from GuiControl.
string GuiCanvas::getMode (int modeId)

Gets information on the specified mode of this device.

**Parameters:**

*modeId* Index of the mode to get data from.

**Returns:**

A video mode string given an adapter and mode index.

**See also:**

GuiCanvas::getVideoMode()

---

int GuiCanvas::getModeCount ( )

Gets the number of modes available on this device.

**Parameters:**

*param* Description

**Example:**

```
%modeCount = Canvas.getModeCount()
```

**Returns:**

The number of video modes supported by the device

---

int GuiCanvas::getMouseDownControl ( )

Gets the gui control under the mouse.

**Example:**

```
%underMouse = Canvas.getMouseDownControl();
```
Returns:

ID of the gui control, if one was found. NULL otherwise

string GuiCanvas::getVideoMode( )

Gets the current screen mode as a string.

The return string will contain 5 values (width, height, fullscreen, bitdepth, refreshRate). You will need to parse out each one for individual use.

Example:

```cpp
%screenWidth = getWord(Canvas.getVideoMode(), 0);
%screenHeight = getWord(Canvas.getVideoMode(), 1);
%isFullscreen = getWord(Canvas.getVideoMode(), 2);
%bitdepth = getWord(Canvas.getVideoMode(), 3);
%refreshRate = getWord(Canvas.getVideoMode(), 4);
```

Returns:

String formatted with screen width, screen height, screen mode, bit depth, and refresh rate.

Point2I GuiCanvas::getWindowPosition( )

Get the current position of the platform window associated with the canvas.

Returns:

The window position of the canvas in screen-space.

void GuiCanvas::hideCursor( )
Disable rendering of the cursor.

Example:

```cpp
Canvas.hideCursor();
```

```cpp
bool GuiCanvas::isCursorOn()
```

Determines if mouse cursor is enabled.

Example:

```cpp
// Is cursor on?
if(Canvas.isCursorOn())
    echo("Canvas cursor is on");
```

Returns:
Returns true if the cursor is on.

```cpp
bool GuiCanvas::isCursorShown()
```

Determines if mouse cursor is rendering.

Example:

```cpp
// Is cursor rendering?
if(Canvas.isCursorShown())
    echo("Canvas cursor is rendering");
```

Returns:
Returns true if the cursor is rendering.

```cpp
bool GuiCanvas::isFullscreen()
```
Is this canvas currently fullscreen?

```cpp
bool GuiCanvas::isMaximized()
```

```cpp
bool GuiCanvas::isMinimized()
```

```cpp
void GuiCanvas::maximizeWindow()
```

maximize this canvas' window.

```cpp
void GuiCanvas::minimizeWindow()
```

minimize this canvas' window.

```cpp
void GuiCanvas::popDialog(GuiControl ctrl)
```

Removes a specific dialog control.

**Parameters:**

`ctrl` Dialog to pop

**Example:**

```cpp
Canvas.popDialog(RecordingsDlg);
```

```cpp
void GuiCanvas::popDialog()
```

Removes a dialog at the front most layer.

**Example:**

```cpp
// Pops whatever is on layer 0
```
void GuiCanvas::popLayer()

Removes the top most layer of dialogs.

Example:

Canvas.popLayer();

void GuiCanvas::popLayer(S32 layer)

Removes a specified layer of dialogs.

Parameters:

layer Number of the layer to pop

Example:

Canvas.popLayer(1);

void GuiCanvas::pushDialog(GuiControl ctrl, int layer = 0, bool center = false)

Adds a dialog control onto the stack of dialogs.

Parameters:

ctrl Dialog to add
layer Layer to put dialog on (optional)
center True to center dialog on canvas (optional)

Example:
Canvas.pushDialog(RecordingsDlg);

void GuiCanvas::renderFront(bool enable)

This turns on/off front-buffer rendering.

**Parameters:**

*enable*  True if all rendering should be done to the front buffer

**Example:**

Canvas.renderFront(false);

void GuiCanvas::repaint(int elapsedMS = 0)

Force canvas to redraw. If the elapsed time is greater than the time since the last paint then the repaint will be skipped.

**Parameters:**

*elapsedMS*  The optional elapsed time in milliseconds.

**Example:**

Canvas.repaint();

void GuiCanvas::reset()

Reset the update regions for the canvas.

**Example:**

Canvas.reset();
void GuiCanvas::restoreWindow()

restore this canvas' window.

Point2I GuiCanvas::screenToClient(Point2I coordinate)

Translate a coordinate from screen-space to canvas window-space.

**Parameters:**

coordinate The coordinate in screen-space.

**Returns:**

The given coordinate translated to window-space.

void GuiCanvas::setContent(GuiControl ctrl)

Set the content of the canvas to a specified control.

**Parameters:**

ctrl ID or name of GuiControl to set content to

**Example:**

Canvas.setContent(PlayGui);

void GuiCanvas::setCursor(GuiCursor cursor)

Sets the cursor for the canvas.

**Parameters:**

cursor Name of the GuiCursor to use
Example:

```cpp
Canvas.setCursor("DefaultCursor");
```

```cpp
bool GuiCanvas::setCursorPos(Point2I pos )
```

Sets the position of the cursor.

**Parameters:**

- `pos` Point, in screenspace for the cursor. Formatted as ("x y")

Example:

```cpp
Canvas.setCursorPos("0 0");
```

```cpp
bool GuiCanvas::setCursorPos(F32 posX,  
              F32 posY  
)  
```

Sets the position of the cursor.

**Parameters:**

- `posX` X-coordinate, in screenspace for the cursor.
- `posY` Y-coordinate, in screenspace for the cursor.

Example:

```cpp
Canvas.setCursorPos(0,0);
```

```cpp
void GuiCanvas::setFocus( )
```

Claim OS input focus for this canvas' window.
void GuiCanvas::setVideoMode(int width, int height, bool fullscreen)

Change the video mode of this canvas. This method has the side effect of setting the $pref::Video::mode to the new values.

**Parameters:**

- **width**  
The screen width to set.
- **height**  
The screen height to set.
- **fullscreen**  
Specify true to run fullscreen or false to run in a window
- **bitDepth**  
[optional] The desired bit-depth. Defaults to the current setting. This parameter is ignored if you are running in a window.
- **refreshRate**  
[optional] The desired refresh rate. Defaults to the current setting. This parameter is ignored if you are running in a window
- **antialiasLevel**  
[optional] The level of anti-aliasing to apply 0 = none

void GuiCanvas::setWindowPosition(Point2I position)

Set the position of the platform window associated with the canvas.

**Parameters:**

- **position**  
The new position of the window in screen-space.

void GuiCanvas::setWindowTitle(string newTitle)

Change the title of the OS window.
Parameters:

newTitle  String containing the new name

Example:

Canvas.setWindowTitle("Documentation Rocks!");

void GuiCanvas::showCursor()

Enable rendering of the cursor.

Example:

Canvas.showCursor();

void GuiCanvas::toggleFullscreen()

toggle canvas from fullscreen to windowed mode or back.

Example:

// If we are in windowed mode, the following:
Canvas.toggleFullscreen();
Member Data Documentation

bool GuiCanvas::alwaysHandleMouseButtons

Deal with mouse buttons, even if the cursor is hidden.

int GuiCanvas::numFences

The number of GFX fences to use.
GuiCheckBoxCtrl Class Reference
[Button Controls]

A named checkbox that can be toggled on and off. More...

Inheritance diagram for GuiCheckBoxCtrl:

```
  SimObject
   |
   v
 SimSet
   |
   v
 SimGroup
   |
   v
 GuiControl
   |
   v
 GuiButtonBaseCtrl
   |
   v
 GuiCheckBoxCtrl
   |
   v
 GuiRadioCtrl
```

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>bool</th>
<th>isStateOn ()</th>
<th>Test whether the checkbox is currently checked.</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>setStateOn (bool newState)</td>
<td>Set whether the checkbox is ticked or not.</td>
</tr>
</tbody>
</table>
Detailed Description

A named checkbox that can be toggled on and off.

A `GuiCheckBoxCtrl` displays a text label next to a checkbox that can be toggled on and off by the user. Checkboxes are usually used to present boolean choices like, for example, a switch to toggle fullscreen video on and off.

Example:

```cpp
// Create a checkbox that allows to toggle fullscreen
new GuiCheckBoxCtrl( FullscreenToggle )
{
    text = "Fullscreen";
};

// Set the initial state to match the current fullscreen setting.
FullscreenToggle.setStateOn( Canvas.isFullscreen() );

// Define function to be called when checkbox state is toggled.
function FullscreenToggle::onClick( %this )
{
    Canvas.toggleFullscreen();
}
```
Member Function Documentation

bool GuiCheckBoxCtrl::isStateOn()

Test whether the checkbox is currently checked.

**Returns:**

True if the checkbox is currently ticked, false otherwise.

void GuiCheckBoxCtrl::setStateOn(bool newState)

Set whether the checkbox is ticked or not.

**Parameters:**

*newState* If true the box will be checked, if false, it will be unchecked.

**Note:**

This method will not trigger the command associated with the control. To toggle the checkbox state as if the user had clicked the control, use `performClick()`.

Reimplemented from `GuiButtonBaseCtrl`.

Copyright © GarageGames, LLC. All Rights Reserved.
GuiChunkedBitmapCtrl Class Reference

[Image and Video Controls]

This is a control that will render a specified bitmap or a bitmap specified in a referenced variable. More...

Inheritance diagram for GuiChunkedBitmapCtrl:

```
          SimObject
            ↓
          SimSet
            ↓
          SimGroup
            ↓
        GuiControl
              ↓
GuiChunkedBitmapCtrl
```

List of all members.
Public Member Functions

void setBitmap (string filename)
Set the image rendered in this control.
**Public Attributes**

**GuiChunkedBitmapCtrl**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filename</td>
<td><strong>bitmap</strong></td>
<td>This is the bitmap to render to the control.</td>
</tr>
<tr>
<td>bool</td>
<td><strong>tile</strong></td>
<td>This is no longer in use.</td>
</tr>
<tr>
<td>bool</td>
<td><strong>useVariable</strong></td>
<td>This decides whether to use the &quot;bitmap&quot; file or a bitmap stored in &quot;variable&quot;.</td>
</tr>
</tbody>
</table>
Detailed Description

This is a control that will render a specified bitmap or a bitmap specified in a referenced variable.

This control allows you to either set a bitmap with the "bitmap" field or with the setBitmap method. You can also choose to reference a variable in the "variable" field such as "$image" and then set "useVariable" to true. This will cause it to synchronize the variable with the bitmap displayed (if the variable holds a valid image). You can then change the variable and effectively changed the displayed image.

Example:

```
$image = "anotherbackground.png";
new GuiChunkedBitmapCtrl(ChunkedBitmap) {
    bitmap = "background.png";
    variable = "$image";
    useVariable = false;
}
// This will result in the control rendering "background.png"
// If we now set the useVariable to true i
ChunkedBitmap.useVariable = true;
```

See also:

GuiControl::variable
Member Function Documentation

**void GuiChunkedBitmapCtrl::setBitmap(string filename )**

Set the image rendered in this control.

**Parameters:**

*filename* The image name you want to set

**Example:**

```
ChunkedBitmap.setBitmap("images/background.png")
```
Member Data Documentation

**filename GuiChunkedBitmapCtrl::bitmap**

This is the bitmap to render to the control.

**bool GuiChunkedBitmapCtrl::tile**

This is no longer in use.

**bool GuiChunkedBitmapCtrl::useVariable**

This decides whether to use the "bitmap" file or a bitmap stored in "variable".

Copyright © GarageGames, LLC. All Rights Reserved.
GuiClockHud Class Reference
[Game Controls]

Basic HUD clock. Displays the current simulation time offset from some base. More...

Inheritance diagram for GuiClockHud:

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>float</th>
<th><code>getTime()</code></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Returns the current time, in seconds.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th><code>setReverseTime(float timeInSeconds=60)</code></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sets a time for a countdown clock.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th><code>setTime(float timeInSeconds=60)</code></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sets the current base time for the clock.</td>
</tr>
</tbody>
</table>
## Public Attributes

### Misc

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorF</td>
<td>fillColor</td>
<td>Standard color for the background of the control.</td>
</tr>
<tr>
<td>ColorF</td>
<td>frameColor</td>
<td>Color for the control's frame.</td>
</tr>
<tr>
<td>bool</td>
<td>showFill</td>
<td>If true, draws a background color behind the control.</td>
</tr>
<tr>
<td>bool</td>
<td>showFrame</td>
<td>If true, draws a frame around the control.</td>
</tr>
<tr>
<td>ColorF</td>
<td>textColor</td>
<td>Color for the text on this control.</td>
</tr>
</tbody>
</table>
Detailed Description

Basic HUD clock. Displays the current simulation time offset from some base.

Example:

```java
new GuiClockHud()
{
    fillColor = "0.0 1.0 0.0 1.0"; // Fills
    frameColor = "1.0 1.0 1.0 1.0"; // Solid
    textColor = "1.0 1.0 1.0 1.0"; // Solid
    showFill = "true";
    showFrame = "true";
};
```
Member Function Documentation

float GuiClockHud::getTime()

Returns the current time, in seconds.

Returns:

timeInseconds Current time, in seconds

Example:

```c++
// Get the current time from the GuiClockHud
%timeInSeconds = %guiClockHud.getTime();
```

void GuiClockHud::setReverseTime(float timeInSeconds = 60)

Sets a time for a countdown clock.

Setting the time like this will cause the clock to count backwards from the specified time.

Parameters:

- `timeInSeconds` Time to set the clock, in seconds (IE: 00:02 would be 120)

See also:

- `setTime`

void GuiClockHud::setTime(float timeInSeconds = 60)

Sets the current base time for the clock.

Parameters:

- `timeInSeconds` Time to set the clock, in seconds (IE: 00:02
Example:

```java
// Define the time, in seconds
%timeInSeconds = 120;

// Change the time on the GuiClockHud control
%guiClockHud.setTime(%timeInSeconds);
```
## Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorF</td>
<td>GuiClockHud::fillColor</td>
<td>Standard color for the background of the control.</td>
</tr>
<tr>
<td>ColorF</td>
<td>GuiClockHud::frameColor</td>
<td>Color for the control's frame.</td>
</tr>
<tr>
<td>bool</td>
<td>GuiClockHud::showFill</td>
<td>If true, draws a background color behind the control.</td>
</tr>
<tr>
<td>bool</td>
<td>GuiClockHud::showFrame</td>
<td>If true, draws a frame around the control.</td>
</tr>
<tr>
<td>ColorF</td>
<td>GuiClockHud::textColor</td>
<td>Color for the text on this control.</td>
</tr>
</tbody>
</table>
GuiConsole Class Reference
[Core Controls]

The on-screen, in-game console. Calls getLog() to get the on-screen console entries, then renders them as needed. More...

Inheritance diagram for GuiConsole:

List of all members.
Public Member Functions

Callbacks

| void onMessageSelected (ConsoleLogEntry::Level level, string message) | Called when a message in the log is clicked. |
Detailed Description

The on-screen, in-game console. Calls getLog() to get the on-screen console entries, then renders them as needed.

Example:

```java
new GuiConsole()
{
   //Properties not specific to this
};
```

See also:

GuiControl
Member Function Documentation

```cpp
void GuiConsole::onMessageSelected(ConsoleLogEntry::Level level, string message)
```

Called when a message in the log is clicked.

**Parameters:**
- `level` Diagnostic level of the message.
- `message` Message text.

Copyright © GarageGames, LLC. All Rights Reserved.
GuiConsoleEditCtrl Class Reference

[Core Controls]

Text entry element of a GuiConsole. More...

Inheritance diagram for GuiConsoleEditCtrl:

List of all members.
Public Attributes

GuiConsoleEditCtrl

bool useSiblingScroller
Detailed Description

Text entry element of a GuiConsole.

Example:

```cpp
new GuiConsoleEditCtrl(ConsoleEntry) {
    profile = "ConsoleTextEditProfile";
    horizSizing = "width";
    vertSizing = "top";
    position = "0 462";
    extent = "640 18";
    minExtent = "8 8";
    visible = "1";
    altCommand = "ConsoleEntry::eval();";
    helpTag = "0";
    maxLength = "255";
    historySize = "40";
    password = "0";
    tabComplete = "0";
    sinkAllKeyEvents = "1";
    useSiblingScroller = "1";
};
```
Member Data Documentation

bool GuiConsoleEditCtrl::useSiblingScroller
GuiContainer Class Reference
[Container Controls]

Brief Desc. More...

Inheritance diagram for GuiContainer:

List of all members.
### Public Attributes

#### Layout

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>anchorBottom</td>
</tr>
<tr>
<td>bool</td>
<td>anchorLeft</td>
</tr>
<tr>
<td>bool</td>
<td>anchorRight</td>
</tr>
<tr>
<td>bool</td>
<td>anchorTop</td>
</tr>
<tr>
<td>GuiDockingType</td>
<td>docking</td>
</tr>
<tr>
<td>RectSpacingI</td>
<td>margin</td>
</tr>
<tr>
<td>RectSpacingI</td>
<td>padding</td>
</tr>
</tbody>
</table>
Detailed Description

Brief Desc.

Example:

```java
// Comment:
%okButton = new ClassObject()
instantiation
```
### Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>GuiContainer::anchorBottom</td>
</tr>
<tr>
<td>bool</td>
<td>GuiContainer::anchorLeft</td>
</tr>
<tr>
<td>bool</td>
<td>GuiContainer::anchorRight</td>
</tr>
<tr>
<td>bool</td>
<td>GuiContainer::anchorTop</td>
</tr>
<tr>
<td>GuiDockingType</td>
<td>GuiContainer::docking</td>
</tr>
<tr>
<td>RectSpacingI</td>
<td>GuiContainer::margin</td>
</tr>
<tr>
<td>RectSpacingI</td>
<td>GuiContainer::padding</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
GuiControl Class Reference
[Core Controls]

Base class for all Gui control objects. More...

Inheritance diagram for GuiControl:
List of all members.
**Public Member Functions**

<table>
<thead>
<tr>
<th>Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td><code>addGuiControl (GuiControl control)</code></td>
<td>Add the given control as a child to this control.</td>
</tr>
<tr>
<td>void</td>
<td><code>clearFirstResponder (bool ignored=false)</code></td>
<td>Clear this control from being the first responder in its hierarchy chain.</td>
</tr>
<tr>
<td>bool</td>
<td><code>controlIsChild (GuiControl control)</code></td>
<td>Test whether the given control is a direct or indirect child to this control.</td>
</tr>
<tr>
<td>GuiControl</td>
<td><code>findHitControl (int x, int y)</code></td>
<td>Find the topmost child control located at the given coordinates.</td>
</tr>
<tr>
<td>string</td>
<td><code>findHitControls (int x, int y, int width, int height)</code></td>
<td>Find all visible child controls that intersect with the given rectangle.</td>
</tr>
<tr>
<td>float</td>
<td><code>getAspect ()</code></td>
<td>Get the aspect ratio of the control's extents.</td>
</tr>
<tr>
<td>Point2I</td>
<td><code>getCenter ()</code></td>
<td>Get the coordinate of the control's center point relative to its parent.</td>
</tr>
<tr>
<td>Point2I</td>
<td><code>getExtent ()</code></td>
<td>Get the width and height of the control.</td>
</tr>
<tr>
<td>GuiControl</td>
<td><code>getFirstResponder ()</code></td>
<td>Get the first responder set on this GuiControl tree.</td>
</tr>
<tr>
<td>Point2I</td>
<td><code>getGlobalCenter ()</code></td>
<td>Get the coordinate of the control's center point in coordinates relative to the root control in its control hierarchy.</td>
</tr>
<tr>
<td>Point2I</td>
<td><code>getGlobalPosition ()</code></td>
<td>Get the position of the control relative to the root of the GuiControl hierarchy it is contained in.</td>
</tr>
<tr>
<td>Class</td>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Point2I</td>
<td>getMinExtent ()</td>
<td>Get the minimum allowed size of the control.</td>
</tr>
<tr>
<td>GuiControl</td>
<td>getParent ()</td>
<td>Get the immediate parent control of the control.</td>
</tr>
<tr>
<td>Point2I</td>
<td>getPosition ()</td>
<td>Get the control's current position relative to its parent.</td>
</tr>
<tr>
<td>GuiCanvas</td>
<td>getRoot ()</td>
<td>Get the canvas on which the control is placed.</td>
</tr>
<tr>
<td>bool</td>
<td>isAwake ()</td>
<td>Test whether the control is currently awake.</td>
</tr>
<tr>
<td>bool</td>
<td>isFirstResponder ()</td>
<td>Test whether the control is the current first responder.</td>
</tr>
<tr>
<td>bool</td>
<td>isMouseLocked ()</td>
<td>Indicates if the mouse is locked in this control.</td>
</tr>
<tr>
<td>bool</td>
<td>isVisible ()</td>
<td>Test whether the control is currently set to be visible.</td>
</tr>
<tr>
<td>void</td>
<td>makeFirstResponder (bool isFirst)</td>
<td></td>
</tr>
<tr>
<td>bool</td>
<td>pointInControl (int x, int y)</td>
<td>Test whether the given point lies within the rectangle of the control.</td>
</tr>
<tr>
<td>void</td>
<td>resize (int x, int y, int width, int height)</td>
<td>Resize and reposition the control using the give coordinates and dimensions. Child controls will resize according to their layout behaviors.</td>
</tr>
<tr>
<td>void</td>
<td>setActive (bool state=true)</td>
<td></td>
</tr>
<tr>
<td>void</td>
<td>setCenter (int x, int y)</td>
<td>Set the control's position by its center point.</td>
</tr>
<tr>
<td>void</td>
<td>setExtent (S32 width, S32 height)</td>
<td>Resize the control to the given dimensions.</td>
</tr>
<tr>
<td>void</td>
<td>setExtent (Point2I p)</td>
<td></td>
</tr>
</tbody>
</table>
Resize the control to the given dimensions.

```cpp
void setFirstResponder ()
Make this control the current first responder.
```

```cpp
void setPosition (int x, int y)
Position the control in the local space of the parent control.
```

```cpp
void setPositionGlobal (int x, int y)
Set position of the control relative to the root of the GuiControl hierarchy it is contained in.
```

```cpp
void setProfile (GuiControlProfile profile)
Set the control profile for the control to use.
```

```cpp
void setValue (string value)
Set the value associated with the control.
```

```cpp
void setVisible (bool state=true)
Set whether the control is visible or not.
```

**Callbacks**

```cpp
void onAction ()
Called when the control's associated action is triggered and no 'command' is defined for the control.
```

```cpp
void onActive (bool state)
Called when the control changes its activeness state, i.e. when going from active to inactive or vice versa.
```

```cpp
void onAdd ()
Called when the control object is registered with the system after the control has been created.
```

```cpp
void onControlDragEnter (GuiControl control, Point2I dropPoint)
Called when a drag&drop operation through GuiDragAndDropControl has entered the control. This is only called for topmost visible controls as the GuiDragAndDropControl moves over them.
```
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void onControlDragExit (GuiControl control, Point2I dropPoint)</td>
<td>Called when a drag&amp;drop operation through GuiDragAndDropControl has exited the control and moved over a different control. This is only called for topmost visible controls as the GuiDragAndDropControl moves off of them.</td>
</tr>
<tr>
<td>void onControlDragged (GuiControl control, Point2I dropPoint)</td>
<td>Called when a drag&amp;drop operation through GuiDragAndDropControl is moving across the control after it has entered it. This is only called for topmost visible controls as the GuiDragAndDropControl moves across them.</td>
</tr>
<tr>
<td>void onControlDropped (GuiControl control, Point2I dropPoint)</td>
<td>Called when a drag&amp;drop operation through GuiDragAndDropControl has completed and is dropping its payload onto the control. This is only called for topmost visible controls as the GuiDragAndDropControl drops its payload on them.</td>
</tr>
<tr>
<td>void onDialogPop ()</td>
<td>Called when the control is removed as a dialog from the canvas.</td>
</tr>
<tr>
<td>void onDialogPush ()</td>
<td>Called when the control is pushed as a dialog onto the canvas.</td>
</tr>
<tr>
<td>void onGainFirstResponder ()</td>
<td>Called when the control gains first responder status on the GuiCanvas.</td>
</tr>
<tr>
<td>void onLoseFirstResponder ()</td>
<td>Called when the control loses first responder status on the GuiCanvas.</td>
</tr>
<tr>
<td>void onRemove ()</td>
<td>Called when the control object is removed from the</td>
</tr>
</tbody>
</table>
system before it is deleted.

<table>
<thead>
<tr>
<th>void onSleep ()</th>
<th>Called when the control is put to sleep.</th>
</tr>
</thead>
<tbody>
<tr>
<td>void onVisible (bool state)</td>
<td>Called when the control changes its visibility state, i.e. when going from visible to invisible or vice versa.</td>
</tr>
<tr>
<td>void onWake ()</td>
<td>Called when the control is woken up.</td>
</tr>
</tbody>
</table>
### Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td><code>getValue</code></td>
<td></td>
</tr>
<tr>
<td>bool</td>
<td><code>isActive</code></td>
<td></td>
</tr>
</tbody>
</table>

### Control

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td><code>accelerator</code></td>
<td>Key combination that triggers the control's primary action when the control is on the canvas.</td>
</tr>
<tr>
<td>bool</td>
<td><code>active</code></td>
<td>Whether the control is enabled for user interaction.</td>
</tr>
<tr>
<td>string</td>
<td><code>altCommand</code></td>
<td>Command to execute on the secondary action of the control.</td>
</tr>
<tr>
<td>string</td>
<td><code>command</code></td>
<td>Command to execute on the primary action of the control.</td>
</tr>
<tr>
<td></td>
<td><code>modal</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>profile</code></td>
<td>The control profile that determines fill styles, font settings, etc.</td>
</tr>
<tr>
<td></td>
<td><code>setFirstResponder</code></td>
<td></td>
</tr>
<tr>
<td>string</td>
<td><code>variable</code></td>
<td>Name of the variable to which the value of this control will be synchronized.</td>
</tr>
<tr>
<td>bool</td>
<td><code>visible</code></td>
<td>Whether the control is visible or hidden.</td>
</tr>
</tbody>
</table>

### Layout
<table>
<thead>
<tr>
<th><strong>Point2I</strong></th>
<th><strong>extent</strong></th>
<th>The width and height of the control.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GuiHorizontalSizing</strong></td>
<td><strong>horizSizing</strong></td>
<td>The horizontal resizing behavior.</td>
</tr>
<tr>
<td><strong>Point2I</strong></td>
<td><strong>minExtent</strong></td>
<td>The minimum width and height of the control. The control will not be resized smaller than this.</td>
</tr>
<tr>
<td><strong>Point2I</strong></td>
<td><strong>position</strong></td>
<td>The position relative to the parent control.</td>
</tr>
<tr>
<td><strong>GuiVerticalSizing</strong></td>
<td><strong>vertSizing</strong></td>
<td>The vertical resizing behavior.</td>
</tr>
</tbody>
</table>

**ToolTip**

<table>
<thead>
<tr>
<th><strong>int</strong></th>
<th><strong>hovertime</strong></th>
<th>Time for mouse to hover over control until tooltip is shown (in milliseconds).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>string</strong></td>
<td><strong>tooltip</strong></td>
<td>String to show in tooltip for this control.</td>
</tr>
<tr>
<td><strong>GuiControlProfile</strong></td>
<td><strong>tooltipProfile</strong></td>
<td>Control profile to use when rendering tooltips for this control.</td>
</tr>
</tbody>
</table>

**Editing**

| **bool**    | **isContainer** | If true, the control may contain child controls. |

**Localization**

| **string** | **langTableMod** | Name of string table to use for lookup of internationalized text. |


Detailed Description

Base class for all Gui control objects.

*GuiControl* is the basis for the Gui system. It represents an individual control that can be placed on the canvas and with which the mouse and keyboard can potentially interact with.
Control Hierarchies

GuiControls are arranged in a hierarchy. All children of a control are placed in their parent's coordinate space, i.e. their coordinates are relative to the upper left corner of their immediate parent. When a control is moved, all its child controls are moved along with it.

Since GuiControl's are SimGroups, hierarchy also implies ownership. This means that if a control is destroyed, all its children are destroyed along with it. It also means that a given control can only be part of a single GuiControl hierarchy. When adding a control to another control, it will automatically be reparented from another control it may have previously been parented to.
Layout System

GuiControls have a two-dimensional position and are rectangular in shape.
Event System

Control Profiles

Common data accessed by GuiControls is stored in so-called "Control Profiles." This includes font, color, and texture information. By pooling this data in shared objects, the appearance of any number of controls can be changed quickly and easily by modifying only the shared profile object.

If not explicitly assigned a profile, a control will by default look for a profile object that matches its class name. This means that the class GuiMyCtrl, for example, will look for a profile called 'GuiMyProfile'. If this profile cannot be found, the control will fall back to GuiDefaultProfile which must be defined in any case for the Gui system to work.

In addition to its primary profile, a control may be assigned a second profile called 'tooltipProfile' that will be used to render tooltip popups for the control.
Triggered Actions

First Responders

At any time, a single control can be what is called the "first responder" on the GuiCanvas is placed on. This control will be the first control to receive keyboard events not bound in the global ActionMap. If the first responder choses to handle a particular keyboard event,
Waking and Sleeping

Visibility and Activeness

By default, a *GuiControl* is active which means that it

See also:

- GuiCanvas
- GuiControlProfile
Member Function Documentation

`void GuiControl::addGuiControl(GuiControl control)`

Add the given control as a child to this control.

This is synonymous to calling `SimGroup::addObject`.

**Parameters:**

- `control` The control to add as a child.

**Note:**

The control will retain its current position and size.

**See also:**

- `SimGroup::addObject`  
  - Control Hierarchies

`void GuiControl::clearFirstResponder(bool ignored = false)`

Clear this control from being the first responder in its hierarchy chain.

**Parameters:**

- `ignored` Ignored. Supported for backwards-compatibility.

`bool GuiControl::controlsChild(GuiControl control)`

Test whether the given control is a direct or indirect child to this control.

**Parameters:**

- `control` The potential child control.

**Returns:**
True if the given control is a direct or indirect child to this control.

```cpp
GuiControl GuiControl::findHitControl(int x,
                                       int y)
```

Find the topmost child control located at the given coordinates.

**Note:**
Only children that are both visible and have the 'modal' flag set in their profile will be considered in the search.

**Parameters:**
- `x` The X coordinate in the control's own coordinate space.
- `y` The Y coordinate in the control's own coordinate space.

**Returns:**
The topmost child control at the given coordinates or the control on which the method was called if no matching child could be found.

**See also:**
- `GuiControlProfile::modal`
- `findHitControls`

```cpp
string GuiControl::findHitControls(int x,
                                   int y,
                                   int width,
                                   int height)
```

Find all visible child controls that intersect with the given rectangle.

**Note:**
Invisible child controls will not be included in the search.

**Parameters:**

- **x** The X coordinate of the rectangle's upper left corner in the control's own coordinate space.
- **y** The Y coordinate of the rectangle's upper left corner in the control's own coordinate space.
- **width** The width of the search rectangle in pixels.
- **height** The height of the search rectangle in pixels.

**Returns:**

A space-separated list of the IDs of all visible control objects intersecting the given rectangle.

**Example:**

```cpp
// Lock all controls in the rectangle at x=10 and y=10 and the extent width=100 and height=100.
foreach $( %ctrl in %this.findHitControls(10, 10, 100, 100) ) %ctrl.setLocked(true);
```

**See also:**

- findHitControl

---

```cpp
float GuiControl::getAspect() { }
```

Get the aspect ratio of the control's extents.

**Returns:**

The width of the control divided by its height.

**See also:**

- getExtent

---

```cpp
Point2I GuiControl::getCenter() { }
```
Get the coordinate of the control's center point relative to its parent.

**Returns:**

The coordinate of the control's center point in parent-relative coordinates.

```cpp
Point2I GuiControl::getExtent()
```

Get the width and height of the control.

**Returns:**

A point structure containing the width of the control in x and the height in y.

Reimplemented in `GuiCanvas`.

```cpp
GuiControl GuiControl::getFirstResponder()
```

Get the first responder set on this `GuiControl` tree.

**Returns:**

The first responder set on the control's subtree.

**See also:**

`isFirstResponder`
`makeFirstResponder`
`setFirstResponder` First Responders

```cpp
Point2I GuiControl::getGlobalCenter()
```

Get the coordinate of the control's center point in coordinates relative to the root control in its control hierarchy.
the center coordinate of the control in root-relative coordinates.

**GuiControl::getGlobalPosition()**

Get the position of the control relative to the root of the GuiControl hierarchy it is contained in.

**Returns:**
- The control's current position in root-relative coordinates.

**GuiControl::getMinExtent()**

Get the minimum allowed size of the control.

**Returns:**
- The minimum size to which the control can be shrunk.

**See also:**
- minExtent

**GuiControl::getParent()**

Get the immediate parent control of the control.

**Returns:**
- The immediate parent GuiControl or 0 if the control is not parented to a GuiControl.

**GuiControl::getPosition()**

Get the control's current position relative to its parent.
**Returns:**

The coordinate of the control in its parent's coordinate space.

---

**GuiCanvas GuiControl::getRoot( )**

Get the canvas on which the control is placed.

**Returns:**

The canvas on which the control's hierarchy is currently placed or 0 if the control is not currently placed on a GuiCanvas.

**See also:**

GuiControl_Hierarchy

---

**bool GuiControl::isAwake( )**

Test whether the control is currently awake.

If a control is awake it means that it is part of the GuiControl hierarchy of a GuiCanvas.

**Returns:**

True if the control is awake. *Waking and Sleeping*

---

**bool GuiControl::isFirstResponder( )**

Test whether the control is the current first responder.

**Returns:**

True if the control is the current first responder.

**See also:**
**makeFirstResponder**
**setFirstResponder First Responders**

```cpp
bool GuiControl::isMouseLocked()
```

Indicates if the mouse is locked in this control.

**Returns:**
True if the mouse is currently locked.

```cpp
bool GuiControl::isVisible()
```

Test whether the control is currently set to be visible.

**Returns:**
True if the control is currently set to be visible.

**Note:**
This method does not tell anything about whether the control is actually visible to the user at the moment.

**Visibility and Activeness**

```cpp
void GuiControl::makeFirstResponder(bool isFirst)
```

```cpp
void GuiControl::onAction()
```

Called when the control's associated action is triggered and no 'command' is defined for the control.

**Triggered Actions**
void GuiControl::onActive(bool state)

Called when the control changes its activeness state, i.e. when going from active to inactive or vice versa.

Parameters:
    stat The new activeness state.

See also:
    isActive
    setActive Visibility and Activeness

void GuiControl::onAdd()

Called when the control object is registered with the system after the control has been created.

void GuiControl::onControlDragEnter(GuiControl control, Point2I dropPoint)

Called when a drag&drop operation through GuiDragAndDropControl has entered the control. This is only called for topmost visible controls as the GuiDragAndDropControl moves over them.

Parameters:
    control The payload of the drag operation.
    dropPoint The point at which the payload would be dropped if it were released now. Relative to the canvas.

void GuiControl::onControlDragExit(GuiControl control, Point2I dropPoint)
Called when a drag&drop operation through \texttt{GuiDragAndDropControl} has exited the control and moved over a different control. This is only called for topmost visible controls as the \texttt{GuiDragAndDropControl} moves off of them.

\textbf{Parameters:}

- \textit{control} \quad The payload of the drag operation.
- \textit{dropPoint} \quad The point at which the payload would be dropped if it were released now. Relative to the canvas.

\begin{verbatim}
void GuiControl::onControlDragged (GuiControl control,
                                          Point2I dropPoint
)
\end{verbatim}

Called when a drag&drop operation through \texttt{GuiDragAndDropControl} is moving across the control after it has entered it. This is only called for topmost visible controls as the \texttt{GuiDragAndDropControl} moves across them.

\textbf{Parameters:}

- \textit{control} \quad The payload of the drag operation.
- \textit{dropPoint} \quad The point at which the payload would be dropped if it were released now. Relative to the canvas.

\begin{verbatim}
void GuiControl::onControlDropped (GuiControl control,
                                         Point2I dropPoint
)
\end{verbatim}

Called when a drag&drop operation through \texttt{GuiDragAndDropControl} has completed and is dropping its payload onto the control. This is only called for topmost visible controls as the \texttt{GuiDragAndDropControl} drops its payload on
them.

**Parameters:**

- *control*  
The control that is being dropped onto this control.
- *dropPoint*  
The point at which the control is being dropped. Relative to the canvas.

```cpp
void GuiControl::onDialogPop()
```

Called when the control is removed as a dialog from the canvas.

**See also:**

- `GuiCanvas::popDialog`

```cpp
void GuiControl::onDialogPush()
```

Called when the control is pushed as a dialog onto the canvas.

**See also:**

- `GuiCanvas::pushDialog`

```cpp
void GuiControl::onGainFirstResponder()
```

Called when the control gains first responder status on the `GuiCanvas`.

**See also:**

- `setFirstResponder`
- `makeFirstResponder`
- `isFirstResponder First Responders`

```cpp
void GuiControl::onLoseFirstResponder()
```
Called when the control loses first responder status on the \texttt{GuiCanvas}.

\textbf{See also:}
\begin{itemize}
  \item \texttt{setFirstResponder}
  \item \texttt{makeFirstResponder}
  \item \texttt{isFirstResponder}
  \item First Responders
\end{itemize}

\begin{verbatim}
void GuiControl::onRemove( )
\end{verbatim}

Called when the control object is removed from the system before it is deleted.

\begin{verbatim}
void GuiControl::onSleep( )
\end{verbatim}

Called when the control is put to sleep.

\textbf{Waking and Sleeping}

\begin{verbatim}
void GuiControl::onVisible(bool state )
\end{verbatim}

Called when the control changes its visibility state, i.e. when going from visible to invisible or vice versa.

\textbf{Parameters:}
\begin{itemize}
  \item \textit{state} The new visibility state.
\end{itemize}

\textbf{See also:}
\begin{itemize}
  \item \texttt{isVisible}
  \item \texttt{setVisible} Visibility and Activeness
\end{itemize}

\begin{verbatim}
void GuiControl::onWake( )
\end{verbatim}
Called when the control is woken up.

**Waking and Sleeping**

```cpp
bool GuiControl::pointInControl(int x, int y)
```

Test whether the given point lies within the rectangle of the control.

**Parameters:**
- `x` X coordinate of the point in parent-relative coordinates.
- `y` Y coordinate of the point in parent-relative coordinates.

**Returns:**
True if the point is within the control, false if not.

**See also:**
- `getExtent`
- `getPosition`

```cpp
void GuiControl::resize(int x, int y, int width, int height)
```

Resize and reposition the control using the give coordinates and dimensions. Child controls will resize according to their layout behaviors.

**Parameters:**
- `x` The new X coordinate of the control in its parent's coordinate space.
**y**  The new Y coordinate of the control in its parent's coordinate space.

**width**  The new width to which the control should be resized.

**height**  The new height to which the control should be resized.

```cpp
void GuiControl::setActive (bool state = true )
```

```cpp
void GuiControl::setCenter (int x, int y )
```

Set the control's position by its center point.

**Parameters:**

- **x**  The X coordinate of the new center point of the control relative to the control's parent.
- **y**  The Y coordinate of the new center point of the control relative to the control's parent.

```cpp
void GuiControl::setExtent (S32 width, S32 height )
```

Resize the control to the given dimensions.

Child controls will resize according to their layout settings.

**Parameters:**

- **width**  The new width of the control in pixels.
- **height**  The new height of the control in pixels.

```cpp
void GuiControl::setExtent (Point2I p )
```
Resize the control to the given dimensions.

Child controls with resize according to their layout settings.

**Parameters:**

\[ p \] The new (width, height) extents of the control.

```cpp
void GuiControl::setFirstResponder()
```

Make this control the current first responder.

**Note:**

Only controls with a profile that has canKeyFocus enabled are able to become first responders.

**See also:**

GuiControlProfile::canKeyFocus

isFirstResponder First Responders

```cpp
void GuiControl::setPosition(int x, int y)
```

Position the control in the local space of the parent control.

**Parameters:**

\[ x \] The new X coordinate of the control relative to its parent's upper left corner.

\[ y \] The new Y coordinate of the control relative to its parent's upper left corner.

```cpp
void GuiControl::setPositionGlobal(int x, int y)
```
Set position of the control relative to the root of the GuiControl hierarchy it is contained in.

**Parameters:**

- \( x \) The new X coordinate of the control relative to the root's upper left corner.
- \( y \) The new Y coordinate of the control relative to the root's upper left corner.

```cpp
void GuiControl::setProfileGuiControlProfile profile )
```

Set the control profile for the control to use.

The profile used by a control determines a great part of its behavior and appearance.

**Parameters:**

- `profile` The new profile the control should use. Control Profiles

```cpp
void GuiControl::setValue (string value )
```

Set the value associated with the control.

**Parameters:**

- `value` The new value for the control.

```cpp
void GuiControl::setVisible (bool state = true )
```

Set whether the control is visible or not.

**Parameters:**
state  The new visibility flag state for the control. Visibility and Activeness
Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td><strong>GuiControl::accelerator</strong></td>
</tr>
<tr>
<td></td>
<td>Key combination that triggers the control's primary action when the control is on the canvas.</td>
</tr>
<tr>
<td>bool</td>
<td><strong>GuiControl::active</strong></td>
</tr>
<tr>
<td></td>
<td>Whether the control is enabled for user interaction.</td>
</tr>
<tr>
<td>string</td>
<td><strong>GuiControl::altCommand</strong></td>
</tr>
<tr>
<td></td>
<td>Command to execute on the secondary action of the control.</td>
</tr>
<tr>
<td>Note:</td>
<td>Within this script snippet, the control on which the altCommand is being executed is bound to the global variable $ThisControl.</td>
</tr>
<tr>
<td>string</td>
<td><strong>GuiControl::command</strong></td>
</tr>
<tr>
<td></td>
<td>Command to execute on the primary action of the control.</td>
</tr>
<tr>
<td>Note:</td>
<td>Within this script snippet, the control on which the command is being executed is bound to the global variable $ThisControl.</td>
</tr>
<tr>
<td>Point2I</td>
<td><strong>GuiControl::extent</strong></td>
</tr>
</tbody>
</table>
The width and height of the control.

**string** *GuiControl::getValue*

**GuiHorizontalSizing** *GuiControl::horizSizing*

The horizontal resizing behavior.

**int** *GuiControl::hovertime*

Time for mouse to hover over control until tooltip is shown (in milliseconds).

**bool** *GuiControl::isActive*

**bool** *GuiControl::isContainer*

If true, the control may contain child controls.

**string** *GuiControl::langTableMod*

Name of string table to use for lookup of internationalized text.

**Point2I** *GuiControl::minExtent*

The minimum width and height of the control. The control will not be resized smaller than this.
<table>
<thead>
<tr>
<th>deprecated GuiControl::modal</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Point2I</strong> GuiControl::position</td>
</tr>
<tr>
<td>The position relative to the parent control.</td>
</tr>
<tr>
<td>GuiControlProfile GuiControl::profile</td>
</tr>
<tr>
<td>The control profile that determines fill styles, font settings, etc.</td>
</tr>
<tr>
<td>deprecated GuiControl::setFirstResponder</td>
</tr>
<tr>
<td><strong>string</strong> GuiControl::tooltip</td>
</tr>
<tr>
<td>String to show in tooltip for this control.</td>
</tr>
<tr>
<td>GuiControlProfile GuiControl::tooltipProfile</td>
</tr>
<tr>
<td>Control profile to use when rendering tooltips for this control.</td>
</tr>
<tr>
<td><strong>string</strong> GuiControl::variable</td>
</tr>
<tr>
<td>Name of the variable to which the value of this control will be synchronized.</td>
</tr>
<tr>
<td>GuiVerticalSizing GuiControl::vertSizing</td>
</tr>
<tr>
<td>The vertical resizing behavior.</td>
</tr>
</tbody>
</table>
**bool GuiControl::visible**

Whether the control is visible or hidden.
GuiControlArrayControl Class Reference
[Container Controls]

Brief Desc. More...

Inheritance diagram for GuiControlArrayControl:

![Inheritance Diagram]

List of all members.
**Public Attributes**

**Array**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>colCount</td>
<td>Number of columns in the array.</td>
</tr>
<tr>
<td>intList</td>
<td>colSizes</td>
<td>Size of each individual column.</td>
</tr>
<tr>
<td>int</td>
<td>colSpacing</td>
<td>Padding to put between columns.</td>
</tr>
<tr>
<td>int</td>
<td>rowSize</td>
<td>Height of a row in the array.</td>
</tr>
<tr>
<td>int</td>
<td>rowSpacing</td>
<td>Padding to put between rows.</td>
</tr>
</tbody>
</table>
Detailed Description

Brief Desc.

Example:

```javascript
// Comment:
%okButton = new ClassObject()
instantiation
```
## Member Data Documentation

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int GuiControlArrayControl::colCount</td>
<td>Number of columns in the array.</td>
</tr>
<tr>
<td>intList GuiControlArrayControl::colSizes</td>
<td>Size of each individual column.</td>
</tr>
<tr>
<td>int GuiControlArrayControl::colSpacing</td>
<td>Padding to put between columns.</td>
</tr>
<tr>
<td>int GuiControlArrayControl::rowSize</td>
<td>Height of a row in the array.</td>
</tr>
<tr>
<td>int GuiControlArrayControl::rowSpacing</td>
<td>Padding to put between rows.</td>
</tr>
</tbody>
</table>
GuiControlProfile Class Reference
[Core Controls]

A collection of properties that determine control behavior and rendering. More...

Inheritance diagram for GuiControlProfile:

List of all members.
Public Member Functions

int getStringWidth (pString)
### Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>category</td>
<td>Category under which the profile will appear in the editor.</td>
</tr>
</tbody>
</table>

### Text

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>autoSizeHeight</td>
<td>Automatically adjust height of control to fit contents.</td>
</tr>
<tr>
<td>bool</td>
<td>autoSizeWidth</td>
<td>Automatically adjust width of control to fit contents.</td>
</tr>
<tr>
<td>ColorI</td>
<td>cursorColor</td>
<td>Color to use for the text cursor.</td>
</tr>
<tr>
<td>GuiFontCharset</td>
<td>fontCharset</td>
<td></td>
</tr>
<tr>
<td>ColorI</td>
<td><em>fontColor</em></td>
<td>Font color for normal text (same as fontColors[0]).</td>
</tr>
<tr>
<td>ColorI</td>
<td><em>fontColorHL</em></td>
<td>Font color for highlighted text (same as fontColors[1]).</td>
</tr>
<tr>
<td>ColorI</td>
<td><em>fontColorLink</em></td>
<td>Font color for links in text (same as fontColors[4]).</td>
</tr>
<tr>
<td>ColorI</td>
<td><em>fontColorLinkHL</em></td>
<td>Font color for highlighted links in text (same as fontColors[5]).</td>
</tr>
<tr>
<td>ColorI</td>
<td><em>fontColorNA</em></td>
<td>Font color when control is not active/disabled (same as fontColors[2]).</td>
</tr>
</tbody>
</table>
### Color

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color</td>
<td><code>fontColors</code></td>
<td>[10] Font colors to use for different text types/states.</td>
</tr>
<tr>
<td>int</td>
<td><code>fontSize</code></td>
<td>Font size in points.</td>
</tr>
<tr>
<td>string</td>
<td><code>fontType</code></td>
<td>Name of font family and typeface (e.g. &quot;Arial Bold&quot;).</td>
</tr>
</tbody>
</table>

### GuiAlignmentType

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>str</td>
<td>justify</td>
<td>Horizontal alignment for text.</td>
</tr>
</tbody>
</table>

### bool

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>numbersOnly</code></td>
<td>Whether control should only accept numerical data (<code>GuiTextEditCtrl</code>).</td>
</tr>
<tr>
<td>bool</td>
<td><code>returnTab</code></td>
<td>Whether to add automatic tab event when return is pressed so focus moves on to next control (<code>GuiTextEditCtrl</code>).</td>
</tr>
</tbody>
</table>

### Point2I

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float2</td>
<td><code>textOffset</code></td>
<td></td>
</tr>
</tbody>
</table>

### Appearance

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color</td>
<td><code>bevelColorHL</code></td>
<td></td>
</tr>
<tr>
<td>Color</td>
<td><code>bevelColorLL</code></td>
<td></td>
</tr>
<tr>
<td>int</td>
<td><code>border</code></td>
<td>Border type (0=no border).</td>
</tr>
<tr>
<td>Color</td>
<td><code>borderColor</code></td>
<td>Color to draw border with.</td>
</tr>
<tr>
<td>Color</td>
<td><code>borderColorHL</code></td>
<td></td>
</tr>
<tr>
<td>Color</td>
<td><code>borderColorNA</code></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>int</td>
<td><code>borderThickness</code></td>
<td>Thickness of border in pixels.</td>
</tr>
<tr>
<td>ColorI</td>
<td><code>fillColor</code></td>
<td></td>
</tr>
<tr>
<td>ColorI</td>
<td><code>fillColorHL</code></td>
<td></td>
</tr>
<tr>
<td>ColorI</td>
<td><code>fillColorNA</code></td>
<td></td>
</tr>
<tr>
<td>ColorI</td>
<td><code>fillColorSEL</code></td>
<td></td>
</tr>
<tr>
<td>bool</td>
<td><code>opaque</code></td>
<td></td>
</tr>
</tbody>
</table>

**Misc**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filename</td>
<td><code>bitmap</code></td>
<td>Texture to use for rendering control.</td>
</tr>
<tr>
<td>bool</td>
<td><code>hasBitmapArray</code></td>
<td>If true, 'bitmap' is an array of images.</td>
</tr>
<tr>
<td>string</td>
<td><code>profileForChildren</code></td>
<td></td>
</tr>
<tr>
<td>SFXTrack</td>
<td><code>soundButtonDown</code></td>
<td>Sound to play when mouse has been pressed on control.</td>
</tr>
<tr>
<td>SFXTrack</td>
<td><code>soundButtonOver</code></td>
<td>Sound to play when mouse is hovering over control.</td>
</tr>
</tbody>
</table>

**Behavior**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>canKeyFocus</code></td>
<td>Whether the control can have the keyboard focus.</td>
</tr>
<tr>
<td>bool</td>
<td><code>modal</code></td>
<td></td>
</tr>
<tr>
<td>bool</td>
<td><code>mouseOverSelected</code></td>
<td></td>
</tr>
<tr>
<td>bool</td>
<td><code>tab</code></td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

A collection of properties that determine control behavior and rendering.
Member Function Documentation

int GuiControlProfile::getStringWidth(pString )
## Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>GuiControlProfile::autoSizeHeight</td>
<td>Automatically adjust height of control to fit contents.</td>
</tr>
<tr>
<td>bool</td>
<td>GuiControlProfile::autoSizeWidth</td>
<td>Automatically adjust width of control to fit contents.</td>
</tr>
<tr>
<td>ColorI</td>
<td>GuiControlProfile::bevelColorHL</td>
<td></td>
</tr>
<tr>
<td>ColorI</td>
<td>GuiControlProfile::bevelColorLL</td>
<td></td>
</tr>
<tr>
<td>filename</td>
<td>GuiControlProfile::bitmap</td>
<td>Texture to use for rendering control.</td>
</tr>
<tr>
<td>int</td>
<td>GuiControlProfile::border</td>
<td>Border type (0=no border).</td>
</tr>
<tr>
<td>ColorI</td>
<td>GuiControlProfile::borderColor</td>
<td></td>
</tr>
<tr>
<td>ColorI</td>
<td>GuiControlProfile::borderColorHL</td>
<td></td>
</tr>
<tr>
<td>Data Type</td>
<td>Field Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>ColorI</td>
<td>GuiControlProfile::borderColorNA</td>
<td>Thickness of border in pixels.</td>
</tr>
<tr>
<td>int</td>
<td>GuiControlProfile::borderThickness</td>
<td>Whether the control can have the keyboard focus.</td>
</tr>
<tr>
<td>ColorI</td>
<td>GuiControlProfile::canKeyFocus</td>
<td>Category under which the profile will appear in the editor.</td>
</tr>
<tr>
<td>string</td>
<td>GuiControlProfile::category</td>
<td>Color to use for the text cursor.</td>
</tr>
<tr>
<td>ColorI</td>
<td>GuiControlProfile::fillColor</td>
<td>Font charset</td>
</tr>
<tr>
<td>ColorI</td>
<td>GuiControlProfile::fillColorHL</td>
<td></td>
</tr>
<tr>
<td>ColorI</td>
<td>GuiControlProfile::fillColorNA</td>
<td></td>
</tr>
<tr>
<td>ColorI</td>
<td>GuiControlProfile::fillColorSEL</td>
<td></td>
</tr>
<tr>
<td>GuiFontCharset</td>
<td>GuiControlProfile::fontCharset</td>
<td></td>
</tr>
<tr>
<td>ColorI</td>
<td>GuiControlProfile::fontColor</td>
<td></td>
</tr>
</tbody>
</table>
Font color for normal text (same as fontColors[0]).

**ColorI** GuiControlProfile::fontColorHL

Font color for highlighted text (same as fontColors[1]).

**ColorI** GuiControlProfile::fontColorLink

Font color for links in text (same as fontColors[4]).

**ColorI** GuiControlProfile::fontColorLinkHL

Font color for highlighted links in text (same as fontColors[5]).

**ColorI** GuiControlProfile::fontColorNA

Font color when control is not active/disabled (same as fontColors[2]).

**ColorI** GuiControlProfile::fontColors[10]

Font colors to use for different text types/states.

**ColorI** GuiControlProfile::fontColorSEL

Font color for selected text (same as fontColors[3]).

**int** GuiControlProfile::fontSize
Font size in points.

<table>
<thead>
<tr>
<th>String</th>
<th>GuiControlProfile::fontType</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of font family and typeface (e.g. &quot;Arial Bold&quot;).</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Boolean</th>
<th>GuiControlProfile::hasBitmapArray</th>
</tr>
</thead>
<tbody>
<tr>
<td>If true, 'bitmap' is an array of images.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GuiAlignmentType</th>
<th>GuiControlProfile::justify</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horizontal alignment for text.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Boolean</th>
<th>GuiControlProfile::modal</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Boolean</th>
<th>GuiControlProfile::mouseOverSelected</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Boolean</th>
<th>GuiControlProfile::numbersOnly</th>
</tr>
</thead>
<tbody>
<tr>
<td>Whether control should only accept numerical data (GuiTextEditCtrl).</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Boolean</th>
<th>GuiControlProfile::opaque</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>String</th>
<th>GuiControlProfile::profileForChildren</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Boolean</th>
<th>GuiControlProfile::returnTab</th>
</tr>
</thead>
</table>
Whether to add automatic tab event when return is pressed so focus moves on to next control (*GuiTextEditCtrl*).

<table>
<thead>
<tr>
<th>SFXTrack GuiControlProfile::soundButtonDown</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sound to play when mouse has been pressed on control.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SFXTrack GuiControlProfile::soundButtonOver</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sound to play when mouse is hovering over control.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool GuiControlProfile::tab</th>
</tr>
</thead>
</table>

| Point2I GuiControlProfile::textOffset |

Copyright © GarageGames, LLC. All Rights Reserved.
GuiCrossHairHud Class Reference

Basic cross hair hud. Reacts to state of control object. Also displays health bar for named objects under the cross hair. More...

Inheritance diagram for GuiCrossHairHud:

```
  SimObject
     
       SimSet
         
           SimGroup
             
               GuiControl
                 
                   GuiBitmapCtrl
                     
                       GuiCrossHairHud
```

List of all members.
## Public Attributes

### Damage

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorF</td>
<td><strong>damageFillColor</strong></td>
</tr>
<tr>
<td></td>
<td>As the health bar depletes, this color will represent the health loss amount.</td>
</tr>
<tr>
<td>ColorF</td>
<td><strong>damageFrameColor</strong></td>
</tr>
<tr>
<td></td>
<td>Color for the health bar's frame.</td>
</tr>
<tr>
<td>Point2I</td>
<td><strong>damageOffset</strong></td>
</tr>
<tr>
<td></td>
<td>Offset for drawing the damage portion of the health control.</td>
</tr>
<tr>
<td>Point2I</td>
<td><strong>damageRect</strong></td>
</tr>
<tr>
<td></td>
<td>Size for the health bar portion of the control.</td>
</tr>
</tbody>
</table>
Detailed Description

Basic cross hair hud. Reacts to state of control object. Also displays health bar for named objects under the cross hair.

Uses the base bitmap control to render a bitmap, and decides whether to draw or not depending on the current control object and it's state. If there is ShapeBase object under the cross hair and it's named, then a small health bar is displayed.

Example:

```java
new GuiCrossHairHud(){
    damageFillColor = "1.0 0.0 0.0 1.0"; // Fills with a solid red color
    damageFrameColor = "1.0 1.0 1.0 1.0"; // Solid white frame color
    damageRect = "15 5";
    damageOffset = "0 -10";
};
```
**Member Data Documentation**

<table>
<thead>
<tr>
<th>ColorF GuiCrossHairHud::damageFillColor</th>
</tr>
</thead>
<tbody>
<tr>
<td>As the health bar depletes, this color will represent the health loss amount.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ColorF GuiCrossHairHud::damageFrameColor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color for the health bar's frame.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Point2I GuiCrossHairHud::damageOffset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offset for drawing the damage portion of the health control.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Point2I GuiCrossHairHud::damageRect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size for the health bar portion of the control.</td>
</tr>
</tbody>
</table>
GuiCursor Class Reference
[Core Controls]

Acts as a skin for the cursor, where each GuiCursor object can have its own look and click-zone. More...

Inheritance diagram for GuiCursor:

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filename</td>
<td>bitmapName</td>
<td>File name of the bitmap for the cursor.</td>
</tr>
<tr>
<td>Point2I hotSpot</td>
<td></td>
<td>The location of the cursor's hot spot (which pixel carries the click).</td>
</tr>
<tr>
<td>Point2F renderOffset</td>
<td></td>
<td>Offset of the bitmap, where 0 signifies left edge of the bitmap, 1, the right. Similarly for the Y-component.</td>
</tr>
</tbody>
</table>
Detailed Description

Acts as a skin for the cursor, where each GuiCursor object can have its own look and click-zone.

GuiCursors act as skins for the cursor in the game, where each individual GuiCursor can have its own defined imagemap, click zone and render offset. This allows a game to easily support a wide range of cursors. The active cursor can be changed for each Canvas using canvasObj.setCursor(GuiCursor);

Example:

```java
new GuiCursor(DefaultCursor) {
    hotSpot = "1 1";
    renderOffset = "0 0";
    bitmapName = "~/art/gui/images/defaultCursor";
};
```

See also:

GuiCanvas
Member Data Documentation

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filename</td>
<td>GuiCursor::bitmapName</td>
<td>File name of the bitmap for the cursor.</td>
</tr>
<tr>
<td>Point2I</td>
<td>GuiCursor::hotSpot</td>
<td>The location of the cursor's hot spot (which pixel carries the click).</td>
</tr>
<tr>
<td>Point2F</td>
<td>GuiCursor::renderOffset</td>
<td>Offset of the bitmap, where 0 signifies left edge of the bitmap, 1, the right. Similarly for the Y-component.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
GuiDirectoryFileListCtrl Class Reference
[General Controls]

A control that displays a list of files from within a single directory in the game file system. More...

Inheritance diagram for GuiDirectoryFileListCtrl:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>string getSelectedFile()</code></td>
<td>Get the currently selected filename.</td>
</tr>
<tr>
<td><code>string getSelectedFiles()</code></td>
<td>Get the list of selected files.</td>
</tr>
<tr>
<td><code>void reload()</code></td>
<td>Update the file list.</td>
</tr>
<tr>
<td><code>void setFilter (string filter)</code></td>
<td>Set the file filter.</td>
</tr>
<tr>
<td><code>bool setPath (string path, string filter)</code></td>
<td>Set the search path and file filter.</td>
</tr>
</tbody>
</table>
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>fileFilter</td>
<td>Tab-delimited list of file name patterns. Only matched files will be displayed.</td>
</tr>
<tr>
<td>string</td>
<td>filePath</td>
<td>Path in game directory from which to list files.</td>
</tr>
</tbody>
</table>
Detailed Description

A control that displays a list of files from within a single directory in the game file system.

Example:

```cpp
new GuiDirectoryFileListCtrl()
{
    filePath = "art/shapes";
    fileFilter = ".dts" TAB ".dae";
    //Properties not specific to this control
};
```
### Member Function Documentation

**string GuiDirectoryFileListCtrl::getSelectedFile( )**

Get the currently selected filename.

**Returns:**

The filename of the currently selected file

**string GuiDirectoryFileListCtrl::getSelectedFiles( )**

Get the list of selected files.

**Returns:**

A space separated list of selected files

**void GuiDirectoryFileListCtrl::reload( )**

Update the file list.

**void GuiDirectoryFileListCtrl::setFilter(string filter)**

Set the file filter.

**Parameters:**

- *filter* Tab-delimited list of file name patterns. Only matched files will be displayed.

**bool GuiDirectoryFileListCtrl::setPath(string path, string filter)**
Set the search path and file filter.

**Parameters:**

- **path** Path in game directory from which to list files.
- **filter** Tab-delimited list of file name patterns. Only matched files will be displayed.
**Member Data Documentation**

**string** GuiDirectoryFileListCtrl::fileFilter

Tab-delimited list of file name patterns. Only matched files will be displayed.

**string** GuiDirectoryFileListCtrl::filePath

*Path* in game directory from which to list files.

Copyright © GarageGames, LLC. All Rights Reserved.
GuiDragAndDropControl Class Reference
[Utility Controls]

A container control that can be used to implement drag&drop behavior. More...

Inheritance diagram for GuiDragAndDropControl:

```
SimObject
   ^
   |  SimSet
   |    SimGroup
   |     GuiControl
   |      GuiDragAndDropControl

[legend]
```

List of all members.
<table>
<thead>
<tr>
<th>Public Member Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>void  <strong>startDragging</strong> (int x=0, int y=0)</td>
</tr>
<tr>
<td>Start the drag operation.</td>
</tr>
</tbody>
</table>
### Public Attributes

<table>
<thead>
<tr>
<th>bool</th>
<th>deleteOnMouseUp</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If true, the control deletes itself when the left mouse button is released.</td>
</tr>
</tbody>
</table>
Detailed Description

A container control that can be used to implement drag&drop behavior.

**GuiDragAndDropControl** is a special control that can be used to allow drag&drop behavior to be implemented where GuiControls may be dragged across the canvas and the dropped on other GuiControls.

To start a drag operation, construct a **GuiDragAndDropControl** and add the control that should be drag&dropped as a child to it. Note that this must be a single child control. To drag multiple controls, wrap them in a new **GuiControl** object as a temporary container.

Then, to initiate the drag, add the **GuiDragAndDropControl** to the canvas and call **startDragging()**. You can optionally supply an offset to better position the **GuiDragAndDropControl** on the mouse cursor.

As the **GuiDragAndDropControl** is then moved across the canvas, it will call the **onControlDragEnter()**, **onControlDragExit()**, **onControlDragged()**, and finally **onControlDropped()** callbacks on the visible topmost controls that it moves across. **onControlDropped()** is called when the mouse button is released and the drag operation thus finished.

Example:

```cpp
// The following example implements drag&drop behavior so that one color swatch may be dragged over the other.
// This code is taken from the stock scripts.

//---------------------------------------------------------------------------------------------
// With this method, we start the operation when the mouse is click-dragged away from a color swatch.
function GuiSwatchButtonCtrl::onMouseDragged
{
```
// First we construct a new temporary swatch button that becomes the payload for our drag operation and give it the properties of the swatch button we want to copy.

%payload = new GuiSwatchButtonCtrl();
%payload.assignFieldsFrom( %this );
%payload.position = "0 0";
%payload.dragSourceControl = %this; // Remember where the drag originated from so that we don’t copy a color swatch onto itself.

// Calculate the offset of the GuiDragAndDropControl from the mouse cursor. Here we center it on the cursor.

%xOffset = getWord( %payload.extent, 0 ) / 2;
%yOffset = getWord( %payload.extent, 1 ) / 2;

// Compute the initial position of the GuiDragAndDrop control on the canvas based on the current mouse cursor position.

%cursorpos = Canvas.getCursorPos();
%xPos = getWord( %cursorpos, 0 ) - %xOffset;
%yPos = getWord( %cursorpos, 1 ) - %yOffset;

// Create the drag control.

%ctrl = new GuiDragAndDropControl()
{
    canSaveDynamicFields = "0";
    Profile = "GuiSolidDefaultProfile";
    HorizSizing = "right";
    VertSizing = "bottom";
    Position = %xPos SPC %yPos;
    extent = %payload.extent;
    MinExtent = "4 4";
    canSave = "1";
}
Visible = "1"
hovertime = "1000"

// Let the GuiDragAndDropControl delete itself on mouse-up. When the drag is aborted,
deleteOnMouseUp = true;

// To differentiate drags, use the namespace hierarchy to classify them.
// This will allow a color swatch drag to tell itself apart from a file drag, for example.
class = "GuiDragAndDropControlType_ColorSwatch";

// Add the temporary color swatch to the drag control.
%ctrl.add( %payload );

// Start drag by adding the drag control to the canvas and then calling startDragging().
Canvas.getContent().add( %ctrl );
%ctrl.startDragging( %xOffset, %yOffset

// This method receives the drop when the mouse button is released over a color swatch control during a drag operation.
function GuiSwatchButtonCtrl::onControlDropped
{
    // Make sure this is a color swatch drag.
    if( !%payload.parentGroup.isInNamespaceHierarchy )
        return;

    // If dropped on same button whence we came, do nothing.
if ( %payload.dragSourceControl == %this )
    return;

// If a swatch button control is dropped onto this control,
// copy it's color.

if ( %payload.isMemberOfClass( "GuiSwatchButtonCtrl" ) )
{
    // If the swatch button is part of a color-type inspector,
    // remember the inspector field so we can set the color
    // through it.

    if ( %this.parentGroup.isMemberOfClass( "GuiSwatchButtonCtrl" ) )
    {
        %this.parentGroup.apply( ColorFloatToInt( %payload.color ) );
    }
    else if ( %this.parentGroup.isMemberOfClass( "ColorEditorCtrl" ) )
    {
        %this.parentGroup.apply( %payload.color );
    }
    else
    {
        %this.setColor( %payload.color );
    }
}

See also:

GuiControl::onControlDragEnter
GuiControl::onControlDragExit
GuiControl::onControlDragged
GuiControl::onControlDropped
Member Function Documentation

```cpp
void GuiDragAndDropControl::startDragging(int x = 0,
                                          int y = 0)
```

Start the drag operation.

**Parameters:**

- `x`: X coordinate for the mouse pointer offset which the drag control should position itself.
- `y`: Y coordinate for the mouse pointer offset which the drag control should position itself.
Member Data Documentation

`bool GuiDragAndDropControl::deleteOnMouseUp`

If true, the control deletes itself when the left mouse button is released.

If at this point, the drag&drop control still contains its payload, it will be deleted along with the control.
GuiDynamicCtrlArrayControl Class Reference
[Container Controls]

A container that arranges children into a grid. More...

Inheritance diagram for GuiDynamicCtrlArrayControl:

```
SimObject
  
SimSet
  
SimGroup
  
GuiControl
    
GuiDynamicCtrlArrayControl
```

List of all members.
Public Member Functions

```c
void refresh ()
    Recalculates the position and size of this control and all its children.
```
### Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>autoCellSize</td>
<td>When true, the cell size is set to the widest/tallest child control.</td>
</tr>
<tr>
<td>int</td>
<td>colCount</td>
<td>Number of columns the child controls have been arranged into. This value is calculated automatically when children are added, removed or resized; writing it directly has no effect.</td>
</tr>
<tr>
<td>int</td>
<td>colSize</td>
<td>Width of each column. If autoCellSize is set, this will be calculated automatically from the widest child control.</td>
</tr>
<tr>
<td>int</td>
<td>colSpacing</td>
<td>Spacing between columns.</td>
</tr>
<tr>
<td>bool</td>
<td>dynamicSize</td>
<td>If true, the width or height of this control will be automatically calculated based on the number of child controls (width if fillRowFirst is false, height if fillRowFirst is true).</td>
</tr>
<tr>
<td>bool</td>
<td>fillRowFirst</td>
<td>Controls whether rows or columns are filled first.</td>
</tr>
<tr>
<td>bool</td>
<td>frozen</td>
<td>When true, the array will not update when new children are added or in response to child resize events. This is useful to prevent unnecessary resizing when adding, removing or resizing a number of child controls.</td>
</tr>
<tr>
<td>RectSpacingI</td>
<td>padding</td>
<td>Padding around the top, bottom, left, and right of this control. This reduces the area available for child controls.</td>
</tr>
<tr>
<td><strong>int</strong> rowCount</td>
<td>Number of rows the child controls have been arranged into. This value is calculated automatically when children are added, removed or resized; writing it directly has no effect.</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>int</strong> rowSize</td>
<td>Height of each row. If autoCellSize is set, this will be calculated automatically from the tallest child control.</td>
<td></td>
</tr>
<tr>
<td><strong>int</strong> rowSpacing</td>
<td>Spacing between rows.</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

A container that arranges children into a grid.

This container maintains a 2D grid of GUI controls. If one is added, deleted, or resized, then the grid is updated. The insertion order into the grid is determined by the internal order of the children (ie. the order of addition).

Children are added to the grid by row or column until they fill the associated GuiDynamicCtrlArrayControl extent (width or height). For example, a GuiDynamicCtrlArrayControl with 15 children, and fillRowFirst set to true may be arranged as follows:

```
1 2 3 4 5 6
7 8 9 10 11 12
13 14 15
```

If dynamicSize were set to true in this case, the GuiDynamicCtrlArrayControl height would be calculated to fit the 3 rows of child controls.

Example:

```java
new GuiDynamicCtrlArrayControl()
{
  colSize = "128";
  rowSize = "18";
  colSpacing = "2";
  rowSpacing = "2";
  frozen = "0";
  autoCellSize = "1";
  fillRowFirst = "1";
  dynamicSize = "1";
  padding = "0 0 0 0";
  //Properties not specific to this control
};
```
Member Function Documentation

void GuiDynamicCtrlArrayControl::refresh()

Recalculates the position and size of this control and all its children.
**Member Data Documentation**

<table>
<thead>
<tr>
<th>Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>GuiDynamicCtrlArrayControl::autoCellSize</td>
<td>When true, the cell size is set to the widest/tallest child control.</td>
</tr>
<tr>
<td>int</td>
<td>GuiDynamicCtrlArrayControl::colCount</td>
<td>Number of columns the child controls have been arranged into. This value is calculated automatically when children are added, removed or resized; writing it directly has no effect.</td>
</tr>
<tr>
<td>int</td>
<td>GuiDynamicCtrlArrayControl::colSize</td>
<td>Width of each column. If autoCellSize is set, this will be calculated automatically from the widest child control.</td>
</tr>
<tr>
<td>int</td>
<td>GuiDynamicCtrlArrayControl::colSpacing</td>
<td>Spacing between columns.</td>
</tr>
<tr>
<td>bool</td>
<td>GuiDynamicCtrlArrayControl::dynamicSize</td>
<td>If true, the width or height of this control will be automatically calculated based on the number of child controls (width if fillRowFirst is false, height if fillRowFirst is true).</td>
</tr>
<tr>
<td>bool</td>
<td>GuiDynamicCtrlArrayControl::fillRowFirst</td>
<td>Controls whether rows or columns are filled first.</td>
</tr>
</tbody>
</table>
If true, controls are added to the grid left-to-right (to fill a row); then rows are added top-to-bottom as shown below:

```
1 2 3 4
5 6 7 8
```

If false, controls are added to the grid top-to-bottom (to fill a column); then columns are added left-to-right as shown below:

```
1 3 5 7
2 4 6 8
```

`bool GuiDynamicCtrlArrayControl::frozen`

When true, the array will not update when new children are added or in response to child resize events. This is useful to prevent unnecessary resizing when adding, removing or resizing a number of child controls.

`RectSpacingI GuiDynamicCtrlArrayControl::padding`

Padding around the top, bottom, left, and right of this control. This reduces the area available for child controls.

`int GuiDynamicCtrlArrayControl::rowCount`

Number of rows the child controls have been arranged into. This value is calculated automatically when children are added, removed or resized; writing it directly has no effect.

`int GuiDynamicCtrlArrayControl::rowSize`

Height of each row. If `autoCellSize` is set, this will be calculated automatically from the tallest child control.
int GuiDynamicCtrlArrayControl::rowSpacing

Spacing between rows.
GuiFadeinBitmapCtrl Class Reference
[Core Controls]

A GUI control which renders a black square over a bitmap image. The black square will fade out, then fade back in after a determined time. This control is especially useful for transitions and splash screens. More...

Inheritance diagram for GuiFadeinBitmapCtrl:

```
SimObject
   ^
   |
SimSet
   |
   v
SimGroup
   |
   v
GuiControl
   |
   v
GuiBitmapCtrl
   |
   v
GuiFadeinBitmapCtrl
```

[legend]

List of all members.
### Public Member Functions

#### Callbacks

<table>
<thead>
<tr>
<th>void</th>
<th><strong>click</strong> ()</th>
<th>Informs the script level that this object received a Click event from the cursor or keyboard.</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td><strong>onDone</strong> ()</td>
<td>Informs the script level that this object has completed is fade cycle.</td>
</tr>
</tbody>
</table>
## Public Attributes

### Fading

<table>
<thead>
<tr>
<th>Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>done</code></td>
<td>Whether the fade cycle has finished running.</td>
</tr>
<tr>
<td>ColorF</td>
<td><code>fadeColor</code></td>
<td>Color to fade in from and fade out to.</td>
</tr>
<tr>
<td>EaseF</td>
<td><code>fadeInEase</code></td>
<td>Easing curve for fade-in.</td>
</tr>
<tr>
<td></td>
<td><code>fadeInTime</code></td>
<td>Milliseconds for the bitmap to fade in.</td>
</tr>
<tr>
<td>EaseF</td>
<td><code>fadeOutEase</code></td>
<td>Easing curve for fade-out.</td>
</tr>
<tr>
<td></td>
<td><code>fadeOutTime</code></td>
<td>Milliseconds for the bitmap to fade out.</td>
</tr>
<tr>
<td>int</td>
<td><code>waitTime</code></td>
<td>Milliseconds to wait after fading in before fading out the bitmap.</td>
</tr>
</tbody>
</table>
Detailed Description

A GUI control which renders a black square over a bitmap image. The black square will fade out, then fade back in after a determined time. This control is especially useful for transitions and splash screens.

Example:

```java
new GuiFadeinBitmapCtrl()
{
    fadeinTime = "1000";
    waitTime = "2000";
    fadeoutTime = "1000";
    done = "1";
    // Additional GUI properties that are
}
```

See also:

`GuiBitmapCtrl`
Member Function Documentation

```cpp
void GuiFadeinBitmapCtrl::click()

Informs the script level that this object received a Click event from the cursor or keyboard.

Example:

```cpp
GuiFadeinBitmapCtrl::click(%this)
{
    // Code to run when click occurs
}
```

See also:

Core Controls

```cpp
void GuiFadeinBitmapCtrl::onDone()

Informs the script level that this object has completed is fade cycle.

Example:

```cpp
GuiFadeinBitmapCtrl::onDone(%this)
{
    // Code to run when the fade cycle occurs
}
```

See also:

Core Controls
### Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>GuiFadeinBitmapCtrl::done</code></td>
</tr>
<tr>
<td></td>
<td>Whether the fade cycle has finished running.</td>
</tr>
<tr>
<td>ColorF</td>
<td><code>GuiFadeinBitmapCtrl::fadeColor</code></td>
</tr>
<tr>
<td></td>
<td>Color to fade in from and fade out to.</td>
</tr>
<tr>
<td>EaseF</td>
<td><code>GuiFadeinBitmapCtrl::fadeInEase</code></td>
</tr>
<tr>
<td></td>
<td>Easing curve for fade-in.</td>
</tr>
<tr>
<td>int</td>
<td><code>GuiFadeinBitmapCtrl::fadeInTime</code></td>
</tr>
<tr>
<td></td>
<td>Milliseconds for the bitmap to fade in.</td>
</tr>
<tr>
<td>EaseF</td>
<td><code>GuiFadeinBitmapCtrl::fadeOutEase</code></td>
</tr>
<tr>
<td></td>
<td>Easing curve for fade-out.</td>
</tr>
<tr>
<td>int</td>
<td><code>GuiFadeinBitmapCtrl::fadeOutTime</code></td>
</tr>
<tr>
<td></td>
<td>Milliseconds for the bitmap to fade out.</td>
</tr>
<tr>
<td>int</td>
<td><code>GuiFadeinBitmapCtrl::waitTime</code></td>
</tr>
<tr>
<td></td>
<td>Milliseconds to wait after fading in before fading out the bitmap.</td>
</tr>
</tbody>
</table>
GuiFrameSetCtrl Class Reference
[Container Controls]

A gui control allowing a window to be subdivided into panes, each of which displays a gui control child of the GuiFrameSetCtrl. More...

Inheritance diagram for GuiFrameSetCtrl:

```
  SimObject
   V
  SimSet
   V
  SimGroup
   V
GuiControl
   V
GuiContainer
   V
GuiFrameSetCtrl

[legend]
```

List of all members.
# Public Member Functions

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void addColumn ()</td>
<td>Add a new column.</td>
</tr>
<tr>
<td>void addRow ()</td>
<td>Add a new row.</td>
</tr>
<tr>
<td>void frameBorder (int index, string state=&quot;dynamic&quot;)</td>
<td>Override the borderEnable setting for this frame.</td>
</tr>
<tr>
<td>void frameMinExtent (int index, int width, int height)</td>
<td>Set the minimum width and height for the frame. It will not be possible for the user to resize the frame smaller than this.</td>
</tr>
<tr>
<td>void frameMovable (int index, string state=&quot;dynamic&quot;)</td>
<td>Override the borderMovable setting for this frame.</td>
</tr>
<tr>
<td>void framePadding (int index, RectSpacingI padding)</td>
<td>Set the padding for this frame. Padding introduces blank space on the inside edge of the frame.</td>
</tr>
<tr>
<td>int getColumnCount ()</td>
<td>Get the number of columns.</td>
</tr>
<tr>
<td>int getColumnOffset (int index)</td>
<td>Get the horizontal offset of a column.</td>
</tr>
<tr>
<td>RectSpacingI getFramePadding (int index)</td>
<td>Get the padding for this frame.</td>
</tr>
<tr>
<td>int getRowCount ()</td>
<td>Get the number of rows.</td>
</tr>
<tr>
<td>int getRowOffset (int index)</td>
<td>Get the vertical offset of a row.</td>
</tr>
<tr>
<td>void removeColumn ()</td>
<td>Remove the last (rightmost) column.</td>
</tr>
<tr>
<td>void removeRow ()</td>
<td>Remove the last (bottom) row.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td><code>void setColumnOffset (int index, int offset)</code></td>
<td>Set the horizontal offset of a column.</td>
</tr>
<tr>
<td><code>void setRowOffset (int index, int offset)</code></td>
<td>Set the vertical offset of a row.</td>
</tr>
<tr>
<td><code>void updateSizes ()</code></td>
<td>Recalculates child control sizes.</td>
</tr>
</tbody>
</table>
### Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>autoBalance</td>
<td>If true, row and column offsets are automatically scaled to match the new extents when the control is resized.</td>
</tr>
<tr>
<td>ColorI</td>
<td>borderColor</td>
<td>Color of interior borders between cells.</td>
</tr>
<tr>
<td>GuiFrameState</td>
<td>borderEnable</td>
<td>Controls whether frame borders are enabled.</td>
</tr>
<tr>
<td>GuiFrameState</td>
<td>borderMovable</td>
<td>Controls whether borders can be dynamically repositioned with the mouse by the user.</td>
</tr>
<tr>
<td>int</td>
<td>borderWidth</td>
<td>Width of interior borders between cells in pixels.</td>
</tr>
<tr>
<td>intList</td>
<td>columns</td>
<td>A vector of column offsets (determines the width of each column).</td>
</tr>
<tr>
<td>int</td>
<td>fudgeFactor</td>
<td>Offset for row and column dividers in pixels.</td>
</tr>
<tr>
<td>intList</td>
<td>rows</td>
<td>A vector of row offsets (determines the height of each row).</td>
</tr>
</tbody>
</table>
Detailed Description

A gui control allowing a window to be subdivided into panes, each of which displays a gui control child of the `GuiFrameSetCtrl`.

Each gui control child will have an associated FrameDetail through which frame-specific details can be assigned. Frame-specific values override the values specified for the entire frameset.

Note that it is possible to have more children than frames, or more frames than children. In the former case, the extra children will not be visible (they are moved beyond the visible extent of the frameset). In the latter case, frames will be empty. For example, if a frameset had two columns and two rows but only three gui control children they would be assigned to frames as follows:

```
 1 | 2
-----
 3 |
```

The last frame would be blank.

Example:

```cpp
new GuiFrameSetCtrl()
{
    columns = "3";
    rows = "2";
    borderWidth = "1";
    borderColor = "128 128 128";
    borderEnable = "dynamic";
    borderMovable = "dynamic";
    autoBalance = "1";
    fudgeFactor = "0";
    //Properties not specific to this control
};
```
## Member Function Documentation

### void GuiFrameSetCtrl::addColumn(

Add a new column.

### void GuiFrameSetCtrl::addRow(

Add a new row.

### void GuiFrameSetCtrl::frameBorder(int index, string state = "dynamic")

Override the `borderEnable` setting for this frame.

**Parameters:**

- `index` Index of the frame to modify
- `state` New borderEnable state: "on", "off" or "dynamic"

### void GuiFrameSetCtrl::frameMinExtent(int index, int width, int height)

Set the minimum width and height for the frame. It will not be possible for the user to resize the frame smaller than this.

**Parameters:**

- `index` Index of the frame to modify
- `width` Minimum width in pixels
- `height` Minimum height in pixels
void GuiFrameSetCtrl::frameMovable(int index, string state = "dynamic")

Override the *borderMovable* setting for this frame.

**Parameters:**

- *index*  Index of the frame to modify
- *state*  New borderEnable state: "on", "off" or "dynamic"

void GuiFrameSetCtrl::framePadding(int index, RectSpacingI padding)

Set the padding for this frame. Padding introduces blank space on the inside edge of the frame.

**Parameters:**

- *index*  Index of the frame to modify
- *padding*  Frame top, bottom, left, and right padding

int GuiFrameSetCtrl::getColumnCount ()

Get the number of columns.

**Returns:**

The number of columns

int GuiFrameSetCtrl::getColumnOffset(int index )

Get the horizontal offset of a column.
Parameters:

\( \text{index} \) Index of the column to query

Returns:

Column offset in pixels

RectSpacingI GuiFrameSetCtrl::getFramePadding(int index)

Get the padding for this frame.

Parameters:

\( \text{index} \) Index of the frame to query

int GuiFrameSetCtrl::getRowCount()

Get the number of rows.

Returns:

The number of rows

int GuiFrameSetCtrl::getRowOffset(int index)

Get the vertical offset of a row.

Parameters:

\( \text{index} \) Index of the row to query

Returns:

Row offset in pixels

void GuiFrameSetCtrl::removeColumn()
Remove the last (rightmost) column.

```cpp
void GuiFrameSetCtrl::removeRow()
```

Remove the last (bottom) row.

```cpp
void GuiFrameSetCtrl::setColumnOffset(int index, int offset)
```

Set the horizontal offset of a column.

Note that column offsets must always be in increasing order, and therefore this offset must be between the offsets of the columns either side.

**Parameters:**

- `index` Index of the column to modify
- `offset` New column offset

```cpp
void GuiFrameSetCtrl::setRowOffset(int index, int offset)
```

Set the vertical offset of a row.

Note that row offsets must always be in increasing order, and therefore this offset must be between the offsets of the rows either side.

**Parameters:**

- `index` Index of the row to modify
- `offset` New row offset
void GuiFrameSetCtrl::updateSizes()

Recalculates child control sizes.
### Member Data Documentation

<table>
<thead>
<tr>
<th>Member</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool <code>GuiFrameSetCtrl::autoBalance</code></td>
<td>If true, row and column offsets are automatically scaled to match the new extents when the control is resized.</td>
</tr>
<tr>
<td>ColorI <code>GuiFrameSetCtrl::borderColor</code></td>
<td>Color of interior borders between cells.</td>
</tr>
<tr>
<td>GuiFrameState <code>GuiFrameSetCtrl::borderEnable</code></td>
<td>Controls whether frame borders are enabled. Frames use this value unless overridden for that frame using <code>ctrl.frameBorder(index)</code></td>
</tr>
<tr>
<td>GuiFrameState <code>GuiFrameSetCtrl::borderMovable</code></td>
<td>Controls whether borders can be dynamically repositioned with the mouse by the user. Frames use this value unless overridden for that frame using <code>ctrl.frameMovable(index)</code></td>
</tr>
<tr>
<td>int <code>GuiFrameSetCtrl::borderWidth</code></td>
<td>Width of interior borders between cells in pixels.</td>
</tr>
<tr>
<td>intList <code>GuiFrameSetCtrl::columns</code></td>
<td></td>
</tr>
</tbody>
</table>
A vector of column offsets (determines the width of each column).

```
int GuiFrameSetCtrl::fudgeFactor
```

Offset for row and column dividers in pixels.

```
intList GuiFrameSetCtrl::rows
```

A vector of row offsets (determines the height of each row).
GuiGameListMenuCtrl Class Reference

[Game Controls]

A base class for cross platform menu controls that are gamepad friendly. More...

Inheritance diagram for GuiGameListMenuCtrl:

```
SimObject
    ↓
SimSet
    ↓
SimGroup
    ↓
GuiControl
    ↓
GuiGameListMenuCtrl
    ↓
GuiGameListOptionsCtrl
```

[legend]

List of all members.
Public Member Functions

<table>
<thead>
<tr>
<th>void</th>
<th>activateRow ()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activates the current row. The script callback of the current row will be called (if it has one).</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>addRow (string label, string callback, int icon=-1, int yPad=0, bool useHighlightIcon=true, bool enabled=true)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a row to the list control.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int</th>
<th>getRowCount ()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gets the number of rows on the control.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>getRowLabel (int row)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gets the label displayed on the specified row.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int</th>
<th>getSelectedRow ()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gets the index of the currently selected row.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>isRowEnabled (int row)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Determines if the specified row is enabled or disabled.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>setRowEnabled (int row, bool enabled)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets a row's enabled status according to the given parameters.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>setRowLabel (int row, string label)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets the label on the given row.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>setSelected (int row)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets the selected row. Only rows that are enabled can be selected.</td>
<td></td>
</tr>
</tbody>
</table>

Callbacks

<table>
<thead>
<tr>
<th>void</th>
<th>onChange ()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Called when the selected row changes.</td>
<td></td>
</tr>
</tbody>
</table>
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>callbackOnA</td>
<td>Script callback when the 'A' button is pressed. 'A' inputs are Keyboard: A, Return, Space; Gamepad: A, Start.</td>
</tr>
<tr>
<td>string</td>
<td>callbackOnB</td>
<td>Script callback when the 'B' button is pressed. 'B' inputs are Keyboard: B, Esc, Backspace, Delete; Gamepad: B, Back.</td>
</tr>
<tr>
<td>string</td>
<td>callbackOnX</td>
<td>Script callback when the 'X' button is pressed. 'X' inputs are Keyboard: X; Gamepad: X.</td>
</tr>
<tr>
<td>string</td>
<td>callbackOnY</td>
<td>Script callback when the 'Y' button is pressed. 'Y' inputs are Keyboard: Y; Gamepad: Y.</td>
</tr>
<tr>
<td>bool</td>
<td>debugRender</td>
<td>Enable debug rendering.</td>
</tr>
</tbody>
</table>
Detailed Description

A base class for cross platform menu controls that are gamepad friendly.

This class is used to build row-based menu GUIs that can be easily navigated using the keyboard, mouse or gamepad. The desired row can be selected using the mouse, or by navigating using the Up and Down buttons.

Example:

```javascript
new GuiGameListMenuCtrl()
{
    debugRender = "0";
    callbackOnA = "applyOptions();";
    callbackOnB = "Canvas.setContent(MainMenuGui);"
    callbackOnX = "";
    callbackOnY = "revertOptions();";
    //Properties not specific to this control
};
```

See also:

GuiGameListMenuProfile
Member Function Documentation

```c++
void GuiGameListMenuCtrl::activateRow()
```

Activates the current row. The script callback of the current row will be called (if it has one).

```c++
void GuiGameListMenuCtrl::addRow(string label,
                                 string callback,
                                 int icon = -1,
                                 int yPad = 0,
                                 bool useHighlightIcon = true,
                                 bool enabled = true)
```

Add a row to the list control.

**Parameters:**

- `label`: The text to display on the row as a label.
- `callback`: Name of a script function to use as a callback when this row is activated.
- `icon`: [optional] Index of the icon to use as a marker.
- `useHighlightIcon`: [optional] Does this row use the highlight icon?
- `enabled`: [optional] If this row is initially enabled.

```c++
int GuiGameListMenuCtrl::getRowCount()
```

Gets the number of rows on the control.
Returns:
(int) The number of rows on the control.

string GuiGameListMenuCtrl::getRowLabel(int row )

Gets the label displayed on the specified row.

Parameters:
row Index of the row to get the label of.

Returns:
The label for the row.

int GuiGameListMenuCtrl::getSelectedRow()

Gets the index of the currently selected row.

Returns:
Index of the selected row.

bool GuiGameListMenuCtrl::isRowEnabled(int row )

Determines if the specified row is enabled or disabled.

Parameters:
row The row to set the enabled status of.

Returns:
True if the specified row is enabled. False if the row is not enabled or the given index was not valid.

void GuiGameListMenuCtrl::onChange()
Called when the selected row changes.

```cpp
void GuiGameListMenuCtrl::setRowEnabled(int row, bool enabled)
```

Sets a row's enabled status according to the given parameters.

**Parameters:**
- `row` The index to check for validity.
- `enabled` Indicate true to enable the row or false to disable it.

```cpp
void GuiGameListMenuCtrl::setRowLabel(int row, string label)
```

Sets the label on the given row.

**Parameters:**
- `row` Index of the row to set the label on.
- `label` Text to set as the label of the row.

```cpp
void GuiGameListMenuCtrl::setSelected(int row)
```

Sets the selected row. Only rows that are enabled can be selected.

**Parameters:**
- `row` Index of the row to set as selected.
Member Data Documentation

string GuiGameListMenuCtrl::callbackOnA

Script callback when the 'A' button is pressed. 'A' inputs are
Keyboard: A, Return, Space; Gamepad: A, Start.

string GuiGameListMenuCtrl::callbackOnB

Script callback when the 'B' button is pressed. 'B' inputs are
Keyboard: B, Esc, Backspace, Delete; Gamepad: B, Back.

string GuiGameListMenuCtrl::callbackOnX

Script callback when the 'X' button is pressed. 'X' inputs are
Keyboard: X; Gamepad: X.

string GuiGameListMenuCtrl::callbackOnY

Script callback when the 'Y' button is pressed. 'Y' inputs are
Keyboard: Y; Gamepad: Y.

bool GuiGameListMenuCtrl::debugRender

Enable debug rendering.
GuiGameListMenuProfile Class Reference

[Game Controls]

A GuiControlProfile with additional fields specific to GuiGameListMenuCtrl. More...

Inheritance diagram for GuiGameListMenuProfile:

- SimObject
- GuiControlProfile
- GuiGameListMenuProfile
- GuiGameListOptionsProfile

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Point2I</th>
<th>hitAreaLowerRight</th>
<th>Position of the lower right corner of the row hit area (relative to row's top left corner).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point2I</td>
<td>hitAreaUpperLeft</td>
<td>Position of the upper left corner of the row hit area (relative to row's top left corner).</td>
</tr>
<tr>
<td>Point2I</td>
<td>iconOffset</td>
<td>Offset from the row's top left corner at which to render the row icon.</td>
</tr>
<tr>
<td>Point2I</td>
<td>rowSize</td>
<td>The base size (&quot;width height&quot;) of a row.</td>
</tr>
</tbody>
</table>
Detailed Description

A **GuiControlProfile** with additional fields specific to **GuiGameListMenuCtrl**.

Example:

```java
new GuiGameListMenuProfile()
{
    hitAreaUpperLeft = "10 2";
    hitAreaLowerRight = "190 18";
    iconOffset = "10 2";
    rowSize = "200 20";
    //Properties not specific to this control
};
```
Member Data Documentation

**Point2I GuiGameListMenuProfile::hitAreaLowerRight**

Position of the lower right corner of the row hit area (relative to row's top left corner).

**Point2I GuiGameListMenuProfile::hitAreaUpperLeft**

Position of the upper left corner of the row hit area (relative to row's top left corner).

**Point2I GuiGameListMenuProfile::icon Offset**

Offset from the row's top left corner at which to render the row icon.

**Point2I GuiGameListMenuProfile::rowSize**

The base size ("width height") of a row.
GuiGameListOptionsCtrl Class Reference
[Game Controls]

A control for showing pages of options that are gamepad friendly.

More...

Inheritance diagram for GuiGameListOptionsCtrl:

```
Legend:

- SimObject
- SimSet
- SimGroup
- GuiControl
- GuiGameListMenuCtrl
- GuiGameListOptionsCtrl

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void</code></td>
<td><code>addRow</code></td>
<td>(string label, string options, bool wrapOptions, string callback, int icon=-1, int yPad=0, bool enabled=true) Add a row to the list control.</td>
</tr>
<tr>
<td><code>string</code></td>
<td><code>getCurrentOption</code></td>
<td>(int row) Gets the text for the currently selected option of the given row.</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><code>selectOption</code></td>
<td>(int row, string option) Set the row's current option to the one specified.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>setOptions</code></td>
<td>(int row, string optionsList) Sets the list of options on the given row.</td>
</tr>
</tbody>
</table>

Detailed Description

A control for showing pages of options that are gamepad friendly.

Each row in this control allows the selection of one value from a set of options using the keyboard, gamepad or mouse. The row is rendered as 2 columns: the first column contains the row label, the second column contains left and right arrows (for mouse picking) and the currently selected value.

See also:

GuiGameListOptionsProfile
Member Function Documentation

```cpp
void GuiGameListOptionsCtrl::addRow (string label,
                                           string options,
                                           bool wrapOptions,
                                           string callback,
                                           int icon = -1,
                                           int yPad = 0,
                                           bool enabled = true
) {
  Add a row to the list control.

  Parameters:

  - **label**: The text to display on the row as a label.
  - **options**: A tab separated list of options.
  - **wrapOptions**: Specify true to allow options to wrap at each end or false to prevent wrapping.
  - **callback**: Name of a script function to use as a callback when this row is activated.
  - **icon** [optional]: Index of the icon to use as a marker.
  - **yPad** [optional]: An extra amount of height padding before the row. Does nothing on the first row.
  - **enabled** [optional]: If this row is initially enabled.
}
```

```cpp
string GuiGameListOptionsCtrl::getCurrentOption (int row ) {
  Gets the text for the currently selected option of the given row.

  Parameters:

  - **row**: Index of the row to get the option from.

  Returns:

  A string representing the text currently displayed as the
selected option on the given row. If there is no such displayed
text then the empty string is returned.

```cpp
bool GuiGameListOptionsCtrl::selectOption (int row,
                                           string option)
```

Set the row's current option to the one specified.

**Parameters:**
- **row**  Index of the row to set an option on.
- **option**  The option to be made active.

**Returns:**
- True if the row contained the option and was set, false otherwise.

```cpp
void GuiGameListOptionsCtrl::setOptions (int row,
                                          string optionsList)
```

Sets the list of options on the given row.

**Parameters:**
- **row**  Index of the row to set options on.
- **optionsList**  A tab separated list of options for the control.
GuiGameListOptionsProfile Class Reference
[Game Controls]

A GuiControlProfile with additional fields specific to GuiGameListOptionsCtrl. More...

Inheritance diagram for GuiGameListOptionsProfile:

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>int</th>
<th>columnSplit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Padding between the leftmost edge of the control, and the row's left arrow.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int</th>
<th>rightPad</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Padding between the rightmost edge of the control and the row's right arrow.</td>
</tr>
</tbody>
</table>
Detailed Description

A GuiControlProfile with additional fields specific to GuiGameListOptionsCtrl.

Example:

```java
new GuiGameListOptionsProfile()
{
    columnSplit = "100";
    rightPad = "4";
    //Properties not specific to this control
}
```
Member Data Documentation

int GuiGameListOptionsProfile::columnSplit

Padding between the leftmost edge of the control, and the row's left arrow.

int GuiGameListOptionsProfile::rightPad

Padding between the rightmost edge of the control and the row's right arrow.
GuiGraphCtrl Class Reference
[Value Controls]

A control that plots one or more curves in a chart. More...

Inheritance diagram for GuiGraphCtrl:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void addAutoPlot</td>
<td>Sets up the given plotting curve to automatically plot the value of the variable with a frequency of \textit{updateFrequency}.</td>
</tr>
<tr>
<td>void addDatum</td>
<td>Add a data point to the plot's curve.</td>
</tr>
<tr>
<td>float getDatum</td>
<td>Get a data point on the given plotting curve.</td>
</tr>
<tr>
<td>void matchScale</td>
<td>Set the scale of all specified plots to the maximum scale among them.</td>
</tr>
<tr>
<td>void removeAutoPlot</td>
<td>Stop automatic variable plotting for the given curve.</td>
</tr>
<tr>
<td>void setGraphType</td>
<td>Change the charting type of the given plotting curve.</td>
</tr>
</tbody>
</table>

where \textit{plotId}, \textit{variable}, \textit{updateFrequency}, \textit{plotId}, \textit{value}, \textit{plotId}, \textit{index}, \textit{plotId1}, \textit{plotId2},..., \textit{plotId}, \textit{GuiGraphType}, \textit{graphType}
## Public Attributes

### Graph

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>centerY</td>
<td>Ratio of where to place the center coordinate of the graph on the Y axis. 0.5=middle height of control.</td>
</tr>
<tr>
<td>ColorF</td>
<td>plotColor</td>
<td>Color to use for the plotting curves in the graph.</td>
</tr>
<tr>
<td>int</td>
<td>plotInterval</td>
<td>Interval between auto-plots of plotVariable for the respective curve (in milliseconds).</td>
</tr>
<tr>
<td>GuiGraphType</td>
<td>plotType</td>
<td>Charting type of the plotting curves.</td>
</tr>
<tr>
<td>string</td>
<td>plotVariable</td>
<td>Name of the variable to automatically plot on the curves. If empty, auto-plotting is disabled for the respective curve.</td>
</tr>
</tbody>
</table>
**Detailed Description**

A control that plots one or more curves in a chart.

Up to 6 individual curves can be plotted in the graph. Each plotted curve can have its own display style including its own charting style (`plotType`) and color (`plotColor`).

The data points on each curve can be added in one of two ways:

- Manually by calling `addDatum()`. This causes new data points to be added to the left end of the plotting curve.
- Automatically by letting the graph plot the values of a variable over time. This is achieved by calling `addAutoPlot` and specifying the variable and update frequency.

**Example:**

```c++
// Create a graph that plots a red polyline graph of the FPS counter in a 1 second (1000 milliseconds) interval.
new GuiGraphCtrl( FPSGraph )
{
    plotType[ 0 ] = "PolyLine";
    plotColor[ 0 ] = "1 0 0";
    plotVariable[ 0 ] = "fps::real";
    plotInterval[ 0 ] = 1000;
};
```

**Note:**

Each curve has a maximum number of 200 data points it can have at any one time. Adding more data points to a curve will cause older data points to be removed.
Member Function Documentation

```cpp
void GuiGraphCtrl::addAutoPlot(int plotId, string variable, int updateFrequency)
```

Sets up the given plotting curve to automatically plot the value of the `variable` with a frequency of `updateFrequency`.

**Parameters:**

- `plotId` Index of the plotting curve. Must be 0<=plotId<6.
- `variable` Name of the global variable.
- `updateFrequency` Frequency with which to add new data points to the plotting curve (in milliseconds).

**Example:**

```cpp
// Plot FPS counter at 1 second intervals.
%graph.addAutoPlot( 0, "fps::real", 1000 );
```

```cpp
void GuiGraphCtrl::addDatum(int plotId, float value)
```

Add a data point to the plot's curve.

**Parameters:**

- `plotId` Index of the plotting curve to which to add the data point. Must be 0<=plotId<6.
- `value` Value of the data point to add to the curve.

**Note:**
Data values are added to the **left** end of the plotting curve. A maximum number of 200 data points can be added to any single plotting curve at any one time. If this limit is exceeded, data points on the right end of the curve are culled.

```c
float GuiGraphCtrl::getDatum ( int plotId,  
    int index  
 )
```

Get a data point on the given plotting curve.

**Parameters:**

- `plotId`: Index of the plotting curve from which to fetch the data point. Must be `0<=plotId<6`.
- `index`: Index of the data point on the curve.

**Returns:**

The value of the data point or -1 if `plotId` or `index` are out of range.

```c
void GuiGraphCtrl::matchScale (int plotID1,  
    int plotID2,  
    ...  
 )
```

Set the scale of all specified plots to the maximum scale among them.

**Parameters:**

- `plotID1`: Index of plotting curve.
- `plotID2`: Index of plotting curve.

```c
void GuiGraphCtrl::removeAutoPlot ( int plotId )
```
Stop automatic variable plotting for the given curve.

**Parameters:**

*plotId*  Index of the plotting curve. Must be 0<=plotId<6.

```cpp
void GuiGraphCtrl::setGraphType(int plotId, GuiGraphType graphType)
```

Change the charting type of the given plotting curve.

**Parameters:**

*plotId*  Index of the plotting curve. Must be 0<=plotId<6.  
*graphType*  Charting type to use for the curve.

**Note:**

Instead of calling this method, you can directly assign to *plotType*. 
**Member Data Documentation**

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>Member Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>GuiGraphCtrl::centerY</td>
<td>Ratio of where to place the center coordinate of the graph on the Y axis. 0.5=middle height of control. This allows to account for graphs that have only positive or only negative data points, for example.</td>
</tr>
<tr>
<td>ColorF</td>
<td>GuiGraphCtrl::plotColor[6]</td>
<td>Color to use for the plotting curves in the graph.</td>
</tr>
<tr>
<td>int</td>
<td>GuiGraphCtrl::plotInterval[6]</td>
<td>Interval between auto-plots of plotVariable for the respective curve (in milliseconds).</td>
</tr>
<tr>
<td>GuiGraphType</td>
<td>GuiGraphCtrl::plotType[6]</td>
<td>Charting type of the plotting curves.</td>
</tr>
<tr>
<td>string</td>
<td>GuiGraphCtrl::plotVariable[6]</td>
<td>Name of the variable to automatically plot on the curves. If empty, auto-plotting is disabled for the respective curve.</td>
</tr>
</tbody>
</table>
GuiHealthBarHud Class Reference

[Game Controls]

A basic health bar. Shows the damage value of the current PlayerObjectType control object. More...

Inheritance diagram for GuiHealthBarHud:

```
SimObject

SimSet

SimGroup

GuiControl

GuiHealthBarHud
```

[legend]

List of all members.
## Public Attributes

### Colors

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorF</td>
<td><code>damageFillColor</code></td>
<td>As the health bar depletes, this color will represent the health loss amount.</td>
</tr>
<tr>
<td>ColorF</td>
<td><code>fillColor</code></td>
<td>Standard color for the background of the control.</td>
</tr>
<tr>
<td>ColorF</td>
<td><code>frameColor</code></td>
<td>Color for the control's frame.</td>
</tr>
</tbody>
</table>

### Misc

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>displayEnergy</code></td>
<td>If true, display the energy value rather than the damage value.</td>
</tr>
<tr>
<td>bool</td>
<td><code>showFill</code></td>
<td>If true, we draw the background color of the control.</td>
</tr>
<tr>
<td>bool</td>
<td><code>showFrame</code></td>
<td>If true, we draw the frame of the control.</td>
</tr>
</tbody>
</table>

### Pulse

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><code>pulseRate</code></td>
<td>Speed at which the control will pulse.</td>
</tr>
<tr>
<td>float</td>
<td><code>pulseThreshold</code></td>
<td>Health level the control must be under before the control will pulse.</td>
</tr>
</tbody>
</table>
**Detailed Description**

A basic health bar. Shows the damage value of the current PlayerObjectType control object.

This gui displays the damage value of the current PlayerObjectType control object. The gui can be set to pulse if the health value drops below a set value. This control only works if a server connection exists and it's control object is a PlayerObjectType. If either of these requirements is false, the control is not rendered.

**Example:**

```java
new GuiHealthBarHud(){
    fillColor = "0.0 1.0 0.0 1.0"; // Fills with a solid green color
    frameColor = "1.0 1.0 1.0 1.0"; // Solid white frame color
    damageFillColor = "1.0 0.0 0.0 1.0"; // Fills with a solid red color
    pulseRate = "500";
    pulseThreshold = "0.25";
    showFill = "true";
    showFrame = "true";
    displayEnergy = "false";
};
```
**Member Data Documentation**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ColorF GuiHealthBarHud::damageFillColor</strong></td>
</tr>
<tr>
<td>As the health bar depletes, this color will represent the health loss amount.</td>
</tr>
<tr>
<td><strong>bool GuiHealthBarHud::displayEnergy</strong></td>
</tr>
<tr>
<td>If true, display the energy value rather than the damage value.</td>
</tr>
<tr>
<td><strong>ColorF GuiHealthBarHud::fillColor</strong></td>
</tr>
<tr>
<td>Standard color for the background of the control.</td>
</tr>
<tr>
<td><strong>ColorF GuiHealthBarHud::frameColor</strong></td>
</tr>
<tr>
<td>Color for the control's frame.</td>
</tr>
<tr>
<td><strong>int GuiHealthBarHud::pulseRate</strong></td>
</tr>
<tr>
<td>Speed at which the control will pulse.</td>
</tr>
<tr>
<td><strong>float GuiHealthBarHud::pulseThreshold</strong></td>
</tr>
<tr>
<td>Health level the control must be under before the control will pulse.</td>
</tr>
<tr>
<td><strong>bool GuiHealthBarHud::showFill</strong></td>
</tr>
</tbody>
</table>
If true, we draw the background color of the control.

<table>
<thead>
<tr>
<th>bool GuiHealthBarHud::showFrame</th>
</tr>
</thead>
</table>

If true, we draw the frame of the control.

Copyright © GarageGames, LLC. All Rights Reserved.
GuiHealthTextHud Class Reference
[Game Controls]

Shows the health or energy value of the current PlayerObjectType control object. More...

Inheritance diagram for GuiHealthTextHud:

```
SimObject
  SimSet
    SimGroup
      GuiControl
        GuiHealthTextHud
[legend]
```

List of all members.
Public Attributes

Colors

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorF</td>
<td>fillColor</td>
<td>Color for the background of the control.</td>
</tr>
<tr>
<td>ColorF</td>
<td>frameColor</td>
<td>Color for the control's frame.</td>
</tr>
<tr>
<td>ColorF</td>
<td>textColor</td>
<td>Color for the text on this control.</td>
</tr>
<tr>
<td>ColorF</td>
<td>warningColor</td>
<td>Color for the text when health is low.</td>
</tr>
</tbody>
</table>

Alert

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>pulseRate</td>
<td>Speed at which the control will pulse.</td>
</tr>
<tr>
<td>float</td>
<td>pulseThreshold</td>
<td>Health level at which to begin pulsing.</td>
</tr>
<tr>
<td>float</td>
<td>warnThreshold</td>
<td>The health level at which to use the warningColor.</td>
</tr>
</tbody>
</table>

View

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>showEnergy</td>
<td>If true, display the energy value rather than the damage value.</td>
</tr>
<tr>
<td>bool</td>
<td>showFill</td>
<td>If true, draw the background.</td>
</tr>
<tr>
<td>bool</td>
<td>showFrame</td>
<td>If true, draw the frame.</td>
</tr>
<tr>
<td>bool</td>
<td>showTrueValue</td>
<td></td>
</tr>
</tbody>
</table>
If true, we don't hardcode maxHealth to 100.
**Detailed Description**

Shows the health or energy value of the current PlayerObjectType control object.

This gui can be configured to display either the health or energy value of the current Player Object. It can use an alternate display color if the health or drops below a set value. It can also be set to pulse if the health or energy drops below a set value. This control only works if a server connection exists and it's control object is a PlayerObjectType. If either of these requirements is false, the control is not rendered.

**Example:**

```java
new GuiHealthTextHud()
{
    fillColor = "0.0 0.0 0.0 0.5"; // Fills with a transparent black color
    frameColor = "1.0 1.0 1.0 1.0"; // Solid white frame color
    textColor = "0.0 1.0 0.0 1.0"; // Solid green text color
    warningColor = "1.0 0.0 0.0 1.0"; // Solid red color, used when damaged
    showFill = "true";
    showFrame = "true";
    showTrueValue = "false";
    showEnergy = "false";
    warnThreshold = "50";
    pulseThreshold = "25";
    pulseRate = "500";
    profile = "GuiBigTextProfile"
};
```
### Member Data Documentation

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ColorF</code></td>
<td>GuiHealthTextHud::fillColor</td>
</tr>
<tr>
<td>Color for the background of the control.</td>
<td></td>
</tr>
<tr>
<td><code>ColorF</code></td>
<td>GuiHealthTextHud::frameColor</td>
</tr>
<tr>
<td>Color for the control's frame.</td>
<td></td>
</tr>
<tr>
<td><code>int</code></td>
<td>GuiHealthTextHud::pulseRate</td>
</tr>
<tr>
<td>Speed at which the control will pulse.</td>
<td></td>
</tr>
<tr>
<td><code>float</code></td>
<td>GuiHealthTextHud::pulseThreshold</td>
</tr>
<tr>
<td>Health level at which to begin pulsing.</td>
<td></td>
</tr>
<tr>
<td><code>bool</code></td>
<td>GuiHealthTextHud::showEnergy</td>
</tr>
<tr>
<td>If true, display the energy value rather than the damage value.</td>
<td></td>
</tr>
<tr>
<td><code>bool</code></td>
<td>GuiHealthTextHud::showFill</td>
</tr>
<tr>
<td>If true, draw the background.</td>
<td></td>
</tr>
<tr>
<td><code>bool</code></td>
<td>GuiHealthTextHud::showFrame</td>
</tr>
<tr>
<td>If true, draw the frame.</td>
<td></td>
</tr>
</tbody>
</table>
bool GuiHealthTextHud::showTrueValue

If true, we don't hardcode maxHealth to 100.

ColorF GuiHealthTextHud::textColor

Color for the text on this control.

ColorF GuiHealthTextHud::warningColor

Color for the text when health is low.

float GuiHealthTextHud::warnThreshold

The health level at which to use the warningColor.
GuiIconButtonCtrl Class Reference
[Core Controls]

Draws the bitmap within a special button control. Only a single bitmap is used and the button will be drawn in a highlighted mode when the mouse hovers over it or when it has been clicked. More...

Inheritance diagram for GuiIconButtonCtrl:

```
 SimObject
  |
  V
 SimSet
  |
  V
 SimGroup
  |
  V
 GuiControl
  |
  V
 GuiButtonBaseCtrl
  |
  V
 GuiButtonCtrl
  |
  V
 GuiIconButtonCtrl
```

Legend

List of all members.
Public Member Functions

```c
void setBitmap (string buttonFilename)
    Set the bitmap to use for the button portion of this control.
```
**Public Attributes**

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>autoSize</code></td>
<td>If true, the text and icon will be automatically sized to the size of the control.</td>
</tr>
<tr>
<td>Point2I</td>
<td><code>buttonMargin</code></td>
<td>Margin area around the button.</td>
</tr>
<tr>
<td>filename</td>
<td><code>iconBitmap</code></td>
<td>Bitmap file for the icon to display on the button.</td>
</tr>
<tr>
<td>GuiIconButtonIconLocation</td>
<td><code>iconLocation</code></td>
<td>Where to place the icon on the control. Options are 0 (None), 1 (Left), 2 (Right), 3 (Center).</td>
</tr>
<tr>
<td>bool</td>
<td><code>makeIconSquare</code></td>
<td>If true, will make sure the icon is square.</td>
</tr>
<tr>
<td>bool</td>
<td><code>sizeIconToButton</code></td>
<td>If true, the icon will be scaled to be the same size as the button.</td>
</tr>
<tr>
<td>GuiIconButtonTextLocation</td>
<td><code>textLocation</code></td>
<td>Where to place the text on the control.</td>
</tr>
<tr>
<td>int</td>
<td><code>textMargin</code></td>
<td>Margin between the icon and the text.</td>
</tr>
</tbody>
</table>
Detailed Description

Draws the bitmap within a special button control. Only a single bitmap is used and the button will be drawn in a highlighted mode when the mouse hovers over it or when it has been clicked.

Example:

```java
new GuiIconButtonCtrl(TestIconButton)
{
    buttonMargin = "4 4";
    iconBitmap = "art/gui/lagIcon.png";
    iconLocation = "Center";
    sizeIconToButton = "0";
    makeIconSquare = "1";
    textLocation = "Bottom";
    textMargin = "-2";
    autoSize = "0";
    text = "Lag Icon";
    textID = ""STR_LAG""
    buttonType = "PushButton";
    profile = "GuiIconButtonProfile";
};
```

See also:

GuiControl
GuiButtonCtrl
Member Function Documentation

void GuiIconButtonCtrl::setBitmap(string buttonFilename)

Set the bitmap to use for the button portion of this control.

Parameters:

buttonFilename Filename for the image

Example:

```
// Define the button filename
%buttonFilename = "pearlButton";

// Inform the GuiIconButtonCtrl control to update its main button graphic to the defined bitmap
%thisGuiIconButtonCtrl.setBitmap(%buttonFilename);
```

See also:

GuiControl
GuiButtonCtrl
### Member Data Documentation

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>GuiIconButtonCtrl::autoSize</td>
<td>If true, the text and icon will be automatically sized to the size of the control.</td>
</tr>
<tr>
<td>Point2I</td>
<td>GuiIconButtonCtrl::buttonMargin</td>
<td>Margin area around the button.</td>
</tr>
<tr>
<td>filename</td>
<td>GuiIconButtonCtrl::iconBitmap</td>
<td>Bitmap file for the icon to display on the button.</td>
</tr>
<tr>
<td>GuiIconButtonIconLocation</td>
<td>GuiIconButtonCtrl::iconLocation</td>
<td>Where to place the icon on the control. Options are 0 (None), 1 (Left), 2 (Right), 3 (Center).</td>
</tr>
<tr>
<td>bool</td>
<td>GuiIconButtonCtrl::makeIconSquare</td>
<td>If true, will make sure the icon is square.</td>
</tr>
<tr>
<td>bool</td>
<td>GuiIconButtonCtrl::sizeIconToButton</td>
<td>If true, the icon will be scaled to be the same size as the button.</td>
</tr>
<tr>
<td>GuiIconButtonTextLocation</td>
<td>GuiIconButtonCtrl::textLocation</td>
<td></td>
</tr>
</tbody>
</table>
Where to place the text on the control.

Options are 0 (None), 1 (Bottom), 2 (Right), 3 (Top), 4 (Left), 5 (Center).

int GuiIconButtonCtrl::textMargin

Margin between the icon and the text.
GuiInputCtrl Class Reference
[Utility Controls]

A control that locks the mouse and reports all keyboard input events to script. More...

Inheritance diagram for GuiInputCtrl:

List of all members.
Public Member Functions

Callbacks

```cpp
void onInputEvent (string device, string action, bool state)
    Callback that occurs when an input is triggered on this control.
```
Detailed Description

A control that locks the mouse and reports all keyboard input events to script.

This is useful for implementing custom keyboard handling code, and most commonly used in Torque for a menu that allows a user to remap their in-game controls.

Example:

```plaintext
new GuiInputCtrl(OptRemapInputCtrl)
{
    lockMouse = "0";
    position = "0 0";
    extent = "64 64";
    minExtent = "8 8";
    horizSizing = "center";
    vertSizing = "bottom";
    profile = "GuiInputCtrlProfile";
    visible = "1";
    active = "1";
    tooltipProfile = "GuiToolTipProfile";
    hovertime = "1000";
    isContainer = "0";
    canSave = "1";
    canSaveDynamicFields = "0";
};
```

See also:

GuiMouseEventCtrl
Callback that occurs when an input is triggered on this control.

**Parameters:**

- **device**
  The device type triggering the input, such as keyboard, mouse, etc

- **action**
  The actual event occurring, such as a key or button

- **state**
  True if the action is being pressed, false if it is being release
GuiListBoxCtrl Class Reference
[Core Controls]

A list of text items. More...

Inheritance diagram for GuiListBoxCtrl:

```
SimObject
  
SimSet
  
SimGroup
  
GuiControl
  
GuiListBoxCtrl
  
GuiDirectoryFileListCtrl
```

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void <strong>addFilteredItem</strong> (string newItem)</td>
<td>Checks if there is an item with the exact text of what is passed in, and if so the item is removed from the list and adds that item's data to the filtered list.</td>
</tr>
<tr>
<td>void <strong>clearItemColor</strong> (int index)</td>
<td>Removes any custom coloring from an item at the defined index id in the list.</td>
</tr>
<tr>
<td>void <strong>clearItems</strong> ()</td>
<td>Clears all the items in the listbox.</td>
</tr>
<tr>
<td>void <strong>clearSelection</strong> ()</td>
<td>Sets all currently selected items to unselected.</td>
</tr>
<tr>
<td>void <strong>deleteItem</strong> (int itemIndex)</td>
<td>Removes the list entry at the requested index id from the control and clears the memory associated with it.</td>
</tr>
<tr>
<td>void <strong>doMirror</strong> ()</td>
<td>Informs the GuiListBoxCtrl object to mirror the contents of the GuiListBoxCtrl stored in the mirrorSet field.</td>
</tr>
<tr>
<td>int <strong>findItemText</strong> (string findText, bool bCaseSensitive=false)</td>
<td>Returns index of item with matching text or -1 if none found.</td>
</tr>
<tr>
<td>int <strong>getItemCount</strong> ()</td>
<td>Returns the number of items in the list.</td>
</tr>
<tr>
<td>string <strong>getItemObject</strong> (int index)</td>
<td>Returns the object associated with an item. This only makes sense if you are mirroring a simset.</td>
</tr>
<tr>
<td>string <strong>getItemText</strong> (int index)</td>
<td>Returns the text of the item at the specified index.</td>
</tr>
<tr>
<td>int <strong>getLastClickItem</strong> ()</td>
<td>Request the item index for the item that was last clicked.</td>
</tr>
<tr>
<td>int <strong>getSelCount</strong> ()</td>
<td></td>
</tr>
</tbody>
</table>
Returns the number of items currently selected.

```c
int getSelectedItem ()
```

Returns the selected items index or -1 if none selected. If multiple selections exist it returns the first selected item.

```c
string getSelectedItems ()
```

Returns a space delimited list of the selected items indexes in the list.

```c
void insertItem (string text, int index)
```

Inserts an item into the list at the specified index and returns the index assigned or -1 on error.

```c
void removeFilteredItem (string itemName)
```

Removes an item of the entered name from the filtered items list.

```c
void setCurSel (int indexId)
```

Sets the currently selected item at the specified index.

```c
void setCurSelRange (int indexStart, int indexStop=999999)
```

Sets the current selection range from index start to stop. If no stop is specified it sets from start index to the end of the list.

```c
void setItemColor (int index, ColorF color)
```

Sets the color of a single list entry at the specified index id.

```c
void setItemText (int index, string newtext)
```

Sets the items text at the specified index.

```c
void setItemTooltip (int index, string text)
```

Set the tooltip text to display for the given list item.

```c
void setMultipleSelection (bool allowMultSelections)
```

Enable or disable multiple selections for this GuiListBoxCtrl object.

```c
void setSelected (int index, bool setSelected=true)
```

Sets the item at the index specified to selected or not.

**Callbacks**
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bool isObjectMirrored (string indexIdString)</code></td>
<td>Checks if a list item at a defined index id is mirrored, and returns the result.</td>
</tr>
<tr>
<td><code>void onClearSelection ()</code></td>
<td>Called whenever a selected item in the list is cleared.</td>
</tr>
<tr>
<td><code>void onDeleteKey ()</code></td>
<td>Called whenever the Delete key on the keyboard has been pressed while in this control.</td>
</tr>
<tr>
<td><code>void onDoubleClick ()</code></td>
<td>Called whenever an item in the list has been double clicked.</td>
</tr>
<tr>
<td><code>void onMouseDragged ()</code></td>
<td>Called whenever the mouse is dragged across the control.</td>
</tr>
<tr>
<td><code>void onMouseUp (string itemHit, string mouseClickCount)</code></td>
<td>Called whenever the mouse has previously been clicked down (onMouseDown) and has now been raised on the control. If an item in the list was hit during the click cycle, then the index id of the clicked object along with how many clicks occurred are passed into the callback.</td>
</tr>
<tr>
<td><code>void onSelect (string index, string itemText)</code></td>
<td>Called whenever an item in the list is selected.</td>
</tr>
<tr>
<td><code>void onUnselect (string index, string itemText)</code></td>
<td>Called whenever a selected item in the list has been unselected.</td>
</tr>
</tbody>
</table>
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>allowMultipleSelections</td>
<td>If true, will allow the selection of multiple items in the listbox.</td>
</tr>
<tr>
<td>bool</td>
<td>colorBullet</td>
<td>If true, colored items will render a colored rectangular bullet next to the item text.</td>
</tr>
<tr>
<td>bool</td>
<td>fitParentWidth</td>
<td>If true, the width of the listbox will match the width of its parent control.</td>
</tr>
<tr>
<td>string</td>
<td>makeNameCallback</td>
<td>A script snippet to control what is displayed in the list for a SimObject. Within this snippet, $ThisControl is bound to the guiListBoxCtrl and $ThisObject to the contained object in question.</td>
</tr>
<tr>
<td>string</td>
<td>mirrorSet</td>
<td>If populated with the name of another GuiListBoxCtrl, then this list box will mirror the contents of the mirrorSet listbox.</td>
</tr>
</tbody>
</table>
Detailed Description

A list of text items.

A list of text items where each individual entry can have its own text value, text color and associated SimObject.

Example:

```cpp
new GuiListBoxCtrl(GuiMusicPlayerMusicList) {
    allowMultipleSelections = "true";
    fitParentWidth = "true";
    mirrorSet = "AnotherGuiListBoxCtrl";
    makeNameCallback = "";
    colorBullet = "1";
    //Properties not specific to this control
};
```

See also:

GuiControl
void GuiListBoxCtrl::addFilteredItem(string newItem)

Checks if there is an item with the exact text of what is passed in, and if so the item is removed from the list and adds that item's data to the filtered list.

Parameters:

itemName Name of the item that we wish to add to the filtered item list of the GuiListBoxCtrl.

Example:

// Define the itemName that we wish to add
%itemName = "This Item Name";

// Add the item name to the filtered item
%thisGuiListBoxCtrl.addFilteredItem(%filteredItemName);

See also:

GuiControl

void GuiListBoxCtrl::clearItemColor(int index)

Removes any custom coloring from an item at the defined index id in the list.

Parameters:

index Index id for the item to clear any custom color from.

Example:

// Define the index id
%index = "4";
// Request the GuiListBoxCtrl object to request the GuiListBoxCtrl object to remove any custom coloring from the defined index entry.
%thisGuiListBoxCtrl.clearItemColor(%index);

See also:
GuiControl

void GuiListBoxCtrl::clearItems( )

Clears all the items in the listbox.

Example:

// Inform the GuiListBoxCtrl object to clear all items from its list.
%thisGuiListBoxCtrl.clearItems();

See also:
GuiControl

void GuiListBoxCtrl::clearSelection( )

Sets all currently selected items to unselected.

Detailed description

Example:

// Inform the GuiListBoxCtrl object to set all of its items to unselected.
%thisGuiListBoxCtrl.clearSelection();

See also:
GuiControl

void GuiListBoxCtrl::deleteItem(int itemIndex )
Removes the list entry at the requested index id from the control and clears the memory associated with it.

**Parameters:**

- *itemIndex*  Index id location to remove the item from.

**Example:**

```c
// Define the index id we want to remove
%itemIndex = "8";

// Inform the GuiListBoxCtrl object to remove the item at the defined index id.
%thisGuiListBoxCtrl.deleteItem(%itemIndex);
```

**See also:**

References

```c
void GuiListBoxCtrl::doMirror( )
```

Informs the *GuiListBoxCtrl* object to mirror the contents of the *GuiListBoxCtrl* stored in the mirrorSet field.

**Example:**

```c
// Inform the object to mirror the object located at %thisGuiListBox.mirrorSet
%thisGuiListBox.doMirror();
```

**See also:**

Core Controls

```c
int GuiListBoxCtrl::findItemText(string findText, bool bCaseSensitive = false)
```

Returns index of item with matching text or -1 if none found.
Parameters:

- **findText**: Text in the list to find.
- **isCaseSensitive**: If true, the search will be case sensitive.

Example:

```c
// Define the text we wish to find in the list.
%findText = "Hickory Smoked Gideon"/n/n
%isCaseSensitive = "false";

// Ask the GuiListBoxCtrl object what item id in the list matches the requested text.
%machingId = %thisGuiListBoxCtrl.findItemText(%findText,%isCaseSensitive);
```

Returns:

Index id of item with matching text or -1 if none found.

See also:

- **GuiControl**

```
int GuiListBoxCtrl::getItemCount( )
```

Returns the number of items in the list.

Example:

```
// Request the number of items in the list.
%listItemCount = %thisGuiListBoxCtrl.getItemCount();
```

Returns:

The number of items in the list.

See also:

- **GuiControl**
string GuiListBoxCtrl::getItemObject(int index)

Returns the object associated with an item. This only makes sense if you are mirroring a simset.

Parameters:

index Index id to request the associated item from.

Example:

// Define the index id
%index = "12";

// Request the item from the GuiListBoxCtrl object
%object = %thisGuiListBoxCtrl.getItemObject(%index);

Returns:

The object associated with the item in the list.

See also:

References
%text = %thisGuiListBoxCtrl.getItemText(%index);

Returns:
The text of the requested index id.

See also:
GuiControl

int GuiListBoxCtrl::getLastClickItem()

Request the item index for the item that was last clicked.

Example:

// Request the item index for the last clicked item
%lastClickedIndex = %thisGuiListBoxCtrl.getLastClickItem();

Returns:
Index id for the last clicked item in the list.

See also:
GuiControl

int GuiListBoxCtrl::getSelCount()

Returns the number of items currently selected.

Example:

// Request the number of currently selected items
%selectedItemCount = %thisGuiListBoxCtrl.getSelCount();

Returns:
Number of currently selected items.
See also:

GuiControl

int GuiListBoxCtrl::getSelectedItemId( )

Returns the selected items index or -1 if none selected. If multiple selections exist it returns the first selected item.

Example:

// Request the index id of the currently selected item
%selectedItemId = %thisGuiListBoxCtrl.getSelectedItem();

Returns:

The selected items index or -1 if none selected.

See also:

GuiControl

string GuiListBoxCtrl::getSelectedItems( )

Returns a space delimited list of the selected items indexes in the list.

Example:

// Request a space delimited list of the selected items indexes
%selectionList = %thisGuiListBoxCtrl.getSelectedItems();

Returns:

Space delimited list of the selected items indexes in the list

See also:

GuiControl
void GuiListBoxCtrl::insertItem(string text, int index)

Inserts an item into the list at the specified index and returns the index assigned or -1 on error.

**Parameters:**

- **text**  Text item to add.
- **index** Index id to insert the list item text at.

**Example:**

```c++
// Define the text to insert
%text = "Secret Agent Gideon";

// Define the index entry to insert the text at
%index = "14";

// In form the GuiListBoxCtrl object to insert the text
%assignedId = %thisGuiListBoxCtrl.insertItem(%text, %index);
```

**Returns:**

If successful will return the index id assigned. If unsuccessful, will return -1.

**See also:**

GuiControl

bool GuiListBoxCtrl::isObjectMirrored(string indexIdString)

Checks if a list item at a defined index id is mirrored, and returns the result.

**Parameters:**
indexIdString  Index id of the list to check.

Example:

```cpp
// Engine has requested of the script level
GuiListBoxCtrl::isObjectMirrored(%this, %indexIdString) {
   // Perform code required to check an
   return %isMirrored;
}
```

Returns:
A boolean value on if the list item is mirrored or not.

See also:
GuiControl

```cpp
void GuiListBoxCtrl::onClearSelection( )
```

Called whenever a selected item in the list is cleared.

Example:

```cpp
// A selected item is cleared, causing the
GuiListBoxCtrl::onClearSelection(%this) {
   // Code to run whenever a selected :
}
```

See also:
GuiControl

```cpp
void GuiListBoxCtrl::onDeleteKey( )
```
Called whenever the Delete key on the keyboard has been pressed while in this control.

Example:

```cpp
// The delete key on the keyboard has been pressed while this control is in focus, causing the callback to occur.
GuiListBoxCtrl::onDeleteKey(%this)
{
    // Code to call whenever the delete
}
```

See also:
GuiControl

`void GuiListBoxCtrl::onDoubleClick( )`

Called whenever an item in the list has been double clicked.

Example:

```cpp
// An item in the list is double clicked, causing the callback to occur.
GuiListBoxCtrl::onDoubleClick(%this)
{
    // Code to run whenever an item in the control
}
```

See also:
GuiControl

`void GuiListBoxCtrl::onMouseDragged( )`

Called whenever the mouse is dragged across the control.

Example:
Mouse is dragged across the control, causing the callback to occur:

```cpp
GuiListBoxCtrl::onMouseDragged(%this)
{
    // Code to run whenever the mouse is dragged
}
```

See also:

GuiControl

Called whenever the mouse has previously been clicked down (onMouseDown) and has now been raised on the control. If an item in the list was hit during the click cycle, then the index id of the clicked object along with how many clicks occurred are passed into the callback.

Detailed description

**Parameters:**

- `itemHit`: Index id for the list item that was hit
- `mouseClickCount`: How many mouse clicks occurred on this list item

**Example:**

```cpp
// Mouse was previously clicked down, and
GuiListBoxCtrl::onMouseUp(%this, %itemHit,
{
    // Code to call whenever the mouse is up
}
```

See also:
GuiControl

```cpp
void GuiListBoxCtrl::onSelect(string index,
                               string itemText
)
```

Called whenever an item in the list is selected.

**Parameters:**

- `index`  Index id for the item in the list that was selected.
- `itemText`  Text for the list item at the index that was selected.

**Example:**

```cpp
// An item in the list is selected, causing
GuiListBoxCtrl::onSelect(%this, %index, %:
{
    // Code to run whenever an item in t
}
```

**See also:**

GuiControl

```cpp
void GuiListBoxCtrl::onUnselect(string index,
                                 string itemText
)
```

Called whenever a selected item in the list has been unselected.

**Parameters:**

- `index`  Index id of the item that was unselected
- `itemText`  Text for the list entry at the index id that was unselected
Example:

```cpp
// A selected item is unselected, causing GuiListBoxCtrl::onUnSelect(%this, %indexId)
{
    // Code to run whenever a selected item is unselected
}
```

See also:

`GuiControl`

```cpp
void GuiListBoxCtrl::removeFilteredItem(string itemName)
```

Removes an item of the entered name from the filtered items list.

**Parameters:**

`itemName` Name of the item to remove from the filtered list.

Example:

```cpp
// Define the itemName that you wish to remove.
%itemName = "This Item Name";

// Remove the itemName from the GuiListBoxCtrl
%thisGuiListBoxCtrl.removeFilteredItem(%itemName);
```

See also:

`GuiControl`

```cpp
void GuiListBoxCtrl::setCurSel(int indexId)
```

Sets the currently selected item at the specified index.

**Parameters:**
indexId Index Id to set selected.

Example:

```c
// Define the index id that we wish to select.
%selectId = "4";

// Inform the GuiListBoxCtrl object to set
%thisGuiListBoxCtrl.setCurSel(%selectId);
```

See also:

GuiControl

```c
void GuiListBoxCtrl::setCurSelRange(int indexStart,
                                    int indexStop = 999999)
```

Sets the current selection range from index start to stop. If no stop is specified it sets from start index to the end of the list.

Parameters:

- `indexStart` Index Id to start selection.
- `indexStop` Index Id to end selection.

Example:

```c
// Set start id
%indexStart = "3";

// Set end id
%indexEnd = "6";

// Request the GuiListBoxCtrl object to set
%thisGuiListBoxCtrl.setCurSelRange(%indexStart,%indexEnd);
```
See also:
  GuiControl

```cpp
gui::GuiListBoxCtrl::setItemColor(int index, ColorF color)
```

Sets the color of a single list entry at the specified index id.

**Parameters:**

- `index` Index id to modify the color of in the list.
- `color` Color value to set the list entry to.

**Example:**

```cpp
// Define the index id value
%index = "5";

// Define the color value
%color = "1.0 0.0 0.0";

// Inform the GuiListBoxCtrl object to change the color of the requested index
%thisGuiListBoxCtrl.setItemColor(%index, %color);
```

See also:
  GuiControl

```cpp
gui::GuiListBoxCtrl::setItemText(int index, string newtext)
```

Sets the items text at the specified index.

**Parameters:**
index    Index id to set the item text at.
newtext  Text to change the list item at index id to.

Example:

```cpp
// Define the index id
%index = "12";

// Define the text to set the list item to
%newtext = "Gideon's Fancy Goggles";

// Inform the GuiListBoxCtrl object to change
%thisGuiListBoxCtrl.setItemText(%index, %newtext);
```

See also:

GuiControl

```cpp
void GuiListBoxCtrl::setItemTooltip(int index, string text)
```

Set the tooltip text to display for the given list item.

Parameters:

index    Index id to change the tooltip text
text     Text for the tooltip.

Example:

```cpp
// Define the index id
%index = "12";

// Define the tooltip text
%tooltip = "Gideon's goggles can see through space and time."

// Inform the GuiListBoxCtrl object to set
```
See also:
GuiControl

**void GuiListBoxCtrl::setMultipleSelection(bool allowMultSelections)**

Enable or disable multiple selections for this GuiListBoxCtrl object.

**Parameters:**

*allowMultSelections*  
Boolean variable to set the use of multiple selections or not.

**Example:**

```c
// Define the multiple selection use state.
%allowMultSelections = "true";

// Set the allow multiple selection state
%thisGuiListBoxCtrl.setSelected(%allowMultSelections);
```

See also:
GuiControl

**void GuiListBoxCtrl::setSelected(int index, bool setSelected = true)**

Sets the item at the index specified to selected or not.

**Parameters:**

*index*  
Item index to set selected or unselected.
**setSelected** Boolean selection state to set the requested item index.

**Example:**

```java
// Define the index
%index = "5";

// Define the selection state
%selected = "true"

// Inform the GuiListBoxCtrl object of the new selection state for the requested index entry.
%thisGuiListBoxCtrl.setSelected(%index, %selected);
```

**See also:**

GuiControl
### Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>GuiListBoxCtrl::allowMultipleSelections</td>
<td>If true, will allow the selection of multiple items in the listbox.</td>
</tr>
<tr>
<td>bool</td>
<td>GuiListBoxCtrl::colorBullet</td>
<td>If true, colored items will render a colored rectangular bullet next to the item text.</td>
</tr>
<tr>
<td>bool</td>
<td>GuiListBoxCtrl::fitParentWidth</td>
<td>If true, the width of the listbox will match the width of its parent control.</td>
</tr>
<tr>
<td>string</td>
<td>GuiListBoxCtrl::makeNameCallback</td>
<td>A script snippet to control what is displayed in the list for a SimObject. Within this snippet, $ThisControl is bound to the guiListBoxCtrl and $ThisObject to the contained object in question.</td>
</tr>
<tr>
<td>string</td>
<td>GuiListBoxCtrl::mirrorSet</td>
<td>If populated with the name of another GuiListBoxCtrl, then this list box will mirror the contents of the mirrorSet listbox.</td>
</tr>
</tbody>
</table>
GuiMenuBar Class Reference
[Core Controls]

GUI Control which displays a horizontal bar with individual drop-down menu items. Each menu item may also have submenu items. More...

Inheritance diagram for GuiMenuBar:

List of all members.
**Public Member Functions**

<table>
<thead>
<tr>
<th>void</th>
<th>addMenu (string menuText, int menuId)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Adds a new menu to the menu bar.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>addMenuItem (string targetMenu, string menuItemText, int menuItemId, string accelerator, int checkGroup=-1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Adds a menu item to the specified menu. The menu argument can be either the text of a menu or its id.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>addSubmenuItem (string menuTarget, string menuItem, string submenuItemText, int submenuItemId, string accelerator, int checkGroup)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Adds a menu item to the specified menu. The menu argument can be either the text of a menu or its id.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>clearMenuItems (string menuTarget)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Removes all the menu items from the specified menu.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>clearMenus (int param1, int param2)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Clears all the menus from the menu bar.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>clearSubmenuItems (string menuTarget, string menuItem)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Removes all the menu items from the specified submenu.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>removeMenu (string menuTarget)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Removes the specified menu from the menu bar.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>removeMenuItem (string menuTarget, string menuItemTarget)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Removes the specified menu item from the menu.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>setCheckmarkBitmapIndex (int bitmapIndex)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sets the menu bitmap index for the check mark image.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>setMenuBitmapIndex (string menuTarget, int bitmapIndex, bool bitmaponly, bool drawborder)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sets the bitmap index for the menu and toggles rendering only the bitmap.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>setMenuItemBitmap (string menuTarget, string menuItemTarget, int bitmapIndex)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Sets the specified menu item bitmap index in the bitmap array. Setting the item's index to -1 will remove any bitmap.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void setMenuItemChecked (string menuTarget, string menuItemTarget, bool checked)</td>
<td>Sets the menu item bitmap to a check mark, which by default is the first element in the bitmap array (although this may be changed with setCheckmarkBitmapIndex()). Any other menu items in the menu with the same check group become unchecked if they are checked.</td>
</tr>
<tr>
<td>void setMenuItemEnable (string menuTarget, string menuItemTarget, bool enabled)</td>
<td>Sets the menu item to enabled or disabled based on the enable parameter. The specified menu and menu item can either be text or ids.</td>
</tr>
<tr>
<td>void setMenuItemSubmenuState (string menuTarget, string menuItem, bool isSubmenu)</td>
<td>Sets the given menu item to be a submenu.</td>
</tr>
<tr>
<td>void setMenuItemText (string menuTarget, string menuItemTarget, string newMenuItemText)</td>
<td>Sets the text of the specified menu item to the new string.</td>
</tr>
<tr>
<td>void setMenuItemVisible (string menuTarget, string menuItemTarget, bool isVisible)</td>
<td>Sets the whether or not to display the specified menu.</td>
</tr>
<tr>
<td>void setMenuMargins (int horizontalMargin, int verticalMargin, int bitmapToTextSpacing)</td>
<td>Sets the menu rendering margins: horizontal, vertical, bitmap spacing.</td>
</tr>
<tr>
<td>void setMenuText (string menuTarget, string newMenuText)</td>
<td>Sets the text of the specified menu to the new string.</td>
</tr>
<tr>
<td>void setMenuVisible (string menuTarget, bool visible)</td>
<td>Sets the whether or not to display the specified menu.</td>
</tr>
<tr>
<td>void setSubmenuItemChecked (string menuTarget, string menuItemTarget, string submenuItemText, bool checked)</td>
<td></td>
</tr>
</tbody>
</table>
Sets the menu item bitmap to a check mark, which by default is the first element in the bitmap array (although this may be changed with `setCheckmarkBitmapIndex()`). Any other menu items in the menu with the same check group become unchecked if they are checked.

**Callbacks**

<table>
<thead>
<tr>
<th>Type</th>
<th>Function Name</th>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td><code>onMenuItemSelect</code></td>
<td>(string menuItemId, string menuText, string menuItemId, string menuItemText)</td>
<td>Called whenever an item in a menu is selected.</td>
</tr>
<tr>
<td>void</td>
<td><code>onMenuSelect</code></td>
<td>(string menuId, string menuText)</td>
<td>Called whenever a menu is selected.</td>
</tr>
<tr>
<td>void</td>
<td><code>onMouseInMenu</code></td>
<td>(bool isInMenu)</td>
<td>Called whenever the mouse enters, or persists is in the menu.</td>
</tr>
<tr>
<td>void</td>
<td><code>onSubmenuSelect</code></td>
<td>(string subMenuId, string subMenuText)</td>
<td>Called whenever a submenu is selected.</td>
</tr>
</tbody>
</table>
Public Attributes

<table>
<thead>
<tr>
<th>int</th>
<th>padding</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Extra padding to add to the bounds of the control.</td>
</tr>
</tbody>
</table>
**Detailed Description**

GUI Control which displays a horizontal bar with individual drop-down menu items. Each menu item may also have submenu items.

**Example:**

```java
new GuiMenuBar(newMenuBar)
{
    Padding = "0";
    //Properties not specific to this control have been omitted from this example.
};

// Add a menu to the menu bar
newMenuBar.addMenu(0,"New Menu");

// Add a menu item to the New Menu
newMenuBar.addMenuItem(0,"New Menu Item",0,

// Add a submenu item to the New Menu Item
newMenuBar.addSubmenuItem(0,1,"New Submenu Item");
```

See also:

GuiTickCtrl
Member Function Documentation

```cpp
void GuiMenuBar::addMenu(string menuText, int menuId)
```

Adds a new menu to the menu bar.

**Parameters:**
- `menuText` Text to display for the new menu item.
- `menuId` ID for the new menu item.

**Example:**

```cpp
// Define the menu text
$menuText = "New Menu";

// Define the menu ID.
$menuId = "2";

// Inform the GuiMenuBar control to add the menu.
%thisGuiMenuBar.addMenu($menuText,$menuId);
```

See also:
- GuiTickCtrl

```cpp
void GuiMenuBar::addMenuItem(string targetMenu = "", string menuText = "", int menuitemId = 0, string accelerator = NULL, int checkGroup = -1)
```

Adds a menu item to the specified menu. The menu argument can
be either the text of a menu or its id.

Parameters:

- **menu** Menu name or menu Id to add the new item to.
- **menuItemText** Text for the new menu item.
- **menuItemId** Id for the new menu item.
- **accelerator** Accelerator key for the new menu item.
- **checkGroup** Check group to include this menu item in.

Example:

```c
// Define the menu we wish to add the item
%targetMenu = "New Menu"; or %menu = "4"

// Define the text for the new menu item
%menuItemText = "Menu Item";

// Define the id for the new menu item
%menuItemId = "3";

// Set the accelerator key to toggle this
%accelerator = "n";

// Define the Check Group that this menu is in
%checkGroup = "4";

// Inform the GuiMenuBar control to add the new menu item
%thisGuiMenuBar.addMenuItem(%menu, %menuItemText, %menuItemId, %accelerator, %checkGroup);
```

See also:

- GuiTickCtrl

```c
void GuiMenuBar::addSubmenuItem(string menuTarget,
```
string menuItem,  
string submenuItemText,  
int    submenuItemId,  
string accelerator,  
int    checkGroup  }

Adds a menu item to the specified menu. The menu argument can be either the text of a menu or its id.

Parameters:

$menuTarget   Menu to affect a submenu in
$menuItem     Menu item to affect
$submenuItemText Text to show for the new submenu
$submenuItemId Id for the new submenu
$accelerator  Accelerator key for the new submenu
$checkGroup   Which check group the new submenu should be in, or -1 for none.

Example:

// Define the menuTarget
%menuTarget = "New Menu"; or %menuTarget

// Define the menuItem
%menuItem = "New Menu Item"; or %menuItem

// Define the text for the new submenu
%submenuItemText = "New Submenu Item";

// Define the id for the new submenu
%submenuItemId = "4";

// Define the accelerator key for the new
%accelerator = "n";
// Define the checkgroup for the new submenu
%checkgroup = "7";

// Request the GuiMenuBar control to add the new submenu
%thisGuiMenuBar.addSubmenuItem(%menuTarget,
See also:
GuiTickCtrl

void GuiMenuBar::clearMenuItems(string menuTarget )

Removes all the menu items from the specified menu.

Parameters:

_menuTarget Menu to remove all items from

Example:

// Define the menuTarget
%menuTarget = "New Menu"; or %menuTarget

// Inform the GuiMenuBar control to clear
%thisGuiMenuBar.clearMenuItems(%menuTarget)

See also:
GuiTickCtrl

void GuiMenuBar::clearMenus(int param1,
    int param2 )

Clears all the menus from the menu bar.
Example:

```
// Inform the GuiMenuBar control to clear
%thisGuiMenuBar.clearMenus();
```

See also:

GuiTickCtrl

```cpp
void GuiMenuBar::clearSubmenuItems(string menuTarget,
                                   string menuItem)
```

Removes all the menu items from the specified submenu.

Parameters:

- `menuTarget` Menu to affect a submenu in
- `menuItem` Menu item to affect

Example:

```cpp
// Define the menuTarget
%menuTarget = "New Menu"; or %menuTarget

// Define the menuItem
%menuItem = "New Menu Item"; or %menuItem

// Inform the GuiMenuBar to remove all submenu items from the defined menu item
%thisGuiMenuBar.clearSubmenuItems(%menuTarget,%menuItem);
```

See also:

GuiControl

```cpp
void GuiMenuBar::onMenuItemSelect(string menuId,
                                  string menuText,
                                  string menuAction,
                                  string menuKey)
```
Called whenever an item in a menu is selected.

**Parameters:**

- **menuId**: Index id of the menu which contains the selected menu item
- **menuText**: Text of the menu which contains the selected menu item
- **menuItemId**: Index id of the selected menu item
- **menuItemText**: Text of the selected menu item

**Example:**

```cpp
// A menu item has been selected, causing the callback to occur.
GuiMenuBar::onMenuItemSelect(%this,%menuId,%menuText,%menuItemId,%menuItemText)
{
    // Code to run when the callback occurs
}
```

See also:

GuiTickCtrl

```cpp
void GuiMenuBar::onMenuSelect(string menuId,
                                string menuText
)
```

Called whenever a menu is selected.

**Parameters:**

- **menuId**: Index id of the clicked menu
- **menuText**: Text of the clicked menu
Example:

```cpp
// A menu has been selected, causing the callback to occur.
GuiMenuBar::onMenuSelect(%this,%menuId,%menuText)
{
    // Code to run when the callback occurs
}
```

See also:

`GuiTickCtrl`

---

```cpp
void GuiMenuBar::onMouseInMenu(bool isInMenu)
```

Called whenever the mouse enters, or persists is in the menu.

**Parameters:**

- `isInMenu`  
  True if the mouse has entered the menu, otherwise is false.

**Note:**

To receive this callback, call `setProcessTicks(true)` on the menu bar.

Example:

```cpp
// Mouse enters or persists within the menu, causing the callback to occur.
GuiMenuBar::onMouseInMenu(%this,%hasLeftMenu)
{
    // Code to run when the callback occurs
}
```

See also:

`GuiTickCtrl`
void GuiMenuBar::onSubmenuSelect(string submenuId, string submenuText)

Called whenever a submenu is selected.

**Parameters:**

- `submenuId` Id of the selected submenu
- `submenuText` Text of the selected submenu

**Example:**

```cpp
GuiMenuBar::onSubmenuSelect(%this,%submenuId,%submenuText)
{
    // Code to run when the callback occurs
}
```

**See also:**

GuiTickCtrl

---

void GuiMenuBar::removeMenu(string menuTarget)

Removes the specified menu from the menu bar.

**Parameters:**

- `menuTarget` Menu to remove from the menu bar

**Example:**

```cpp
// Define the menuTarget
%menuTarget = "New Menu"; or %menuTarget

// Inform the GuiMenuBar to remove the defined menu
%thisGuiMenuBar.removeMenu(%menuTarget);
```
See also:  
  GuiTickCtrl

```cpp
void GuiMenuBar::removeMenuItem(string menuTarget,
                                string menuItemTarget)
```

Removes the specified menu item from the menu.

**Parameters:**

- `menuTarget` Menu to affect the menu item in
- `menuItem` Menu item to affect

**Example:**

```cpp
// Define the menuTarget
$menuTarget = "New Menu";  or $menuTarget

// Define the menuItem
$menuItem = "New Menu Item";  or $menuItem

// Request the GuiMenuBar control to remove
%thisGuiMenuBar.removeMenuItem(%menuTarget)
```

See also:  
  GuiTickCtrl

```cpp
void GuiMenuBar::setCheckmarkBitmapIndex(int bitmapIndex)
```

Sets the menu bitmap index for the check mark image.

**Parameters:**

- `bitmapIndex` Bitmap index for the check mark image.
Example:

```cpp
// Define the bitmap index
%bitmapIndex = "2";

// Inform the GuiMenuBar control of the proper bitmap index for the check mark image
%thisGuiMenuBar.setCheckmarkBitmapIndex(%bitmapIndex);
```

See also:

GuiTickCtrl

```cpp
void GuiMenuBar::setMenuBitmapIndex(string menuTarget, int bitmapindex, bool bitmaponly, bool drawborder)
```

Sets the bitmap index for the menu and toggles rendering only the bitmap.

Parameters:

- `menuTarget` Menu to affect
- `bitmapindex` Bitmap index to set for the menu
- `bitmaponly` If true, only the bitmap will be rendered
- `drawborder` If true, a border will be drawn around the menu.

Example:

```cpp
// Define the menuTarget to affect
%menuTarget = "New Menu"; or %menuTarget

// Set the bitmap index
%bitmapIndex = "5";

// Set if we are only to render the bitmap
```
%bitmaponly = "true";

// Set if we are rendering a border or not
%drawborder = "true";

// Inform the GuiMenuBar of the bitmap and
%thisGuiMenuBar.setMenuBitmapIndex(%menuTarget, %bitmapIndex, %bitmapOnly, %drawborder);

See also:
GuiTickCtrl

void GuiMenuBar::setMenuItemBitmap(string menuTarget, string menuItemTarget, int bitmapIndex)

Sets the specified menu item bitmap index in the bitmap array. Setting the item's index to -1 will remove any bitmap.

Parameters:

- **menuTarget**  Menu to affect the menuitem in
- **menuItem**  Menu item to affect
- **bitmapIndex**  Bitmap index to set the menu item to

Example:

```
// Define the menuTarget
%menuTarget = "New Menu";  or  %menuTarget

// Define the menuItem"
%menuItem = "New Menu Item";  or  %menuItem

// Define the bitmapIndex
%bitmapIndex = "6";
```
Inform the GuiMenuBar control to set the menu item to the defined bitmap:

```
%thisGuiMenuBar.setMenuItemBitmap(%menuTarget,%menuItem,%bitmapIndex);
```

See also:

GuiTickCtrl

```c
void GuiMenuBar::setMenuItemChecked(string menuTarget, string menuItemTarget, bool checked)
```

Sets the menu item bitmap to a check mark, which by default is the first element in the bitmap array (although this may be changed with `setCheckmarkBitmapIndex()`). Any other menu items in the menu with the same check group become unchecked if they are checked.

**Parameters:**

- `menuTarget` Menu to work in
- `menuItem` Menu item to affect
- `checked` Whether we are setting it to checked or not

**Example:**

<table>
<thead>
<tr>
<th>menuTarget</th>
<th>menuItem</th>
<th>checked</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns:**

If not void, return value and description

See also:

References

```c
void GuiMenuBar::setMenuItemEnable (string menuTarget, string menuItemTarget,
```
bool enabled
)

sets the menu item to enabled or disabled based on the enable parameter. The specified menu and menu item can either be text or ids.

Detailed description

**Parameters:**

- **menuTarget**
  - Menu to work in
- **menuItemTarget**
  - The menu item inside of the menu to enable or disable
- **enabled**
  - Boolean enable / disable value.

**Example:**

```c
// Define the menu
%menu = "New Menu"; or %menu = "4";

// Define the menu item
%menuItem = "New Menu Item"; or %menuItem = "New Menu Item 4";

// Define the enabled state
%enabled = "true";

// Inform the GuiMenuBar control to set the enabled state of the requested menu item
%thisGuiMenuBar.setMenuItemEnable(%menu, %menuItemTarget, %menuTarget, %menuItem, %enabled)
```

See also:

- GuiTickCtrl

```c
void GuiMenuBar::setMenuItemSubmenuState(string menuTarget, string menuItem, bool isSubmenu)
```
Sets the given menu item to be a submenu.

**Parameters:**

- `menuTarget` Menu to affect a submenu in
- `menuItem` Menu item to affect
- `isSubmenu` Whether or not the menuItem will become a submenu or not

**Example:**

```c
// Define the menuTarget
%menuTarget = "New Menu";  // or %menuTarget

// Define the menuItem
%menuItem = "New Menu Item";  // or %menuItem

// Define whether or not the Menu Item is a submenu
%isSubmenu = "true";

// Inform the GuiMenuBar control to set the defined menu item to be a submenu or not.
%thisGuiMenuBar.setMenuItemSubmenuState(%menuTarget, %menuItem, %isSubmenu);
```

**See also:**

- GuiTickCtrl

---

```c
void GuiMenuBar::setMenuItemText(string menuTarget,  
                                string menuItemTarget,  
                                string newMenuItemText  
)
```

Sets the text of the specified menu item to the new string.

**Parameters:**
menuTarget  Menu to affect
menuItem    Menu item in the menu to change the text
newMenuItemText New menu text

Example:

// Define the menuTarget
%menuTarget = "New Menu"; or %menuTarget

// Define the menuItem
%menuItem = "New Menu Item"; or %menuItem

// Define the new text for the menu item
%newMenuItemText = "Very New Menu Item"

// Inform the GuiMenuBar control to change
%thisGuiMenuBar.setMenuItemText(%menuTarget, %menuItem, %newMenuItemText);

See also:
GuiTickCtrl

void GuiMenuBar::setMenuItemVisible(string menuTarget, 
                                        string menuItemTarget, 
                                        bool isVisible
                                      )

Brief Description.

Detailed description

Parameters:

  menuTarget  Menu to affect the menu item in
  menuItem    Menu item to affect
  isVisible   Visible state to set the menu item to.
Example:

```c++
// Define the menuTarget
%menuTarget = "New Menu"; or %menuTarget

// Define the menuItem
%menuItem = "New Menu Item"; or %menuItem

// Define the visibility state
%isVisible = "true";

// Inform the GuiMenuBarControl of the visibility state
%thisGuiMenuBar.setMenuItemVisible(%menuTarget,%menuItem,%isVisible);
```

See also:

GuiTickCtrl

```c++
void GuiMenuBar::setMenuMargins(int horizontalMargin,
                                 int verticalMargin,
                                 int bitmapToTextSpacing)
```

Sets the menu rendering margins: horizontal, vertical, bitmap spacing.

Detailed description

**Parameters:**

- `horizontalMargin`: Number of pixels on the left and right side of a menu's text.
- `verticalMargin`: Number of pixels on the top and bottom of a menu's text.
- `bitmapToTextSpacing`: Number of pixels between a menu's bitmap and text.
Example:

```c++
// Define the horizontalMargin
%horizontalMargin = "5";

// Define the verticalMargin
%verticalMargin = "5";

// Define the bitmapToTextSpacing
%bitmapToTextSpacing = "12";

// Inform the GuiMenuBar control to set it
%thisGuiMenuBar.setMenuMargins(%horizontalMargin, %verticalMargin, %bitmapToTextSpacing);
```

See also:

GuiTickCtrl

```c++
void GuiMenuBar::setMenuText(string menuTarget, string newMenuText)
```

Sets the text of the specified menu to the new string.

**Parameters:**

- `menuTarget` Menu to affect
- `newMenuText` New menu text

**Example:**

```c++
// Define the menu to affect
%menu = "New Menu";

// Define the text to change the menu to
%newMenuText = "Still a New Menu";
```
// Inform the GuiMenuBar control to change the defined menu to the defined text
%thisGuiMenuBar.setMenuText(%menu,%newMenuText);

See also:
GuiTickCtrl

void GuiMenuBar::setMenuVisible(string menuTarget, bool visible)

Sets the whether or not to display the specified menu.

Parameters:

  menuTarget  Menu item to affect
  visible     Whether the menu item will be visible or not

Example:

// Define the menu to work with
%menuTarget = "New Menu"; or %menuTarget = "Other Menu"

// Define if the menu should be visible or not
%visible = "true"

// Inform the GuiMenuBar control of the new visibility state for the defined menu
%thisGuiMenuBar.setMenuVisible(%menuTarget,%visible);

See also:
GuiTickCtrl

void GuiMenuBar::setSubmenuItemChecked(string menuTarget, string menuItemTarget, string submenuItemText,
Sets the menu item bitmap to a check mark, which by default is the first element in the bitmap array (although this may be changed with `setCheckmarkBitmapIndex()`). Any other menu items in the menu with the same check group become unchecked if they are checked.

**Parameters:**

- `menuTarget` Menu to affect a submenu in
- `menuItem` Menu item to affect
- `submenuItemText` Text to show for submenu
- `checked` Whether or not this submenu item will be checked.

**Example:**

```c
// Define the menuTarget
%menuTarget = "New Menu";  or  %menuTarget

// Define the menuItem
%menuItem = "New Menu Item";  or  %menuItem

// Define the text for the new submenu
%submenuItemText = "Submenu Item";

// Define if this submenu item should be (checked = "true"
%checked = "true";

// Inform the GuiMenuBar control to set the
%thisGuiMenuBar.setSubmenuItemChecked(%menuTarget,%menuItem,%submenuItemText,%checked);
```

**Returns:**

If not void, return value and description
See also:

References
Member Data Documentation

int GuiMenuBar::padding

Extra padding to add to the bounds of the control.
GuiMessageVectorCtrl Class Reference
[Utility Controls]

A chat HUD control that displays messages from a MessageVector.
More...

Inheritance diagram for GuiMessageVectorCtrl:

```
SimObject
   ↓
SimSet
   ↓
SimGroup
   ↓
GuiControl
   ↓
GuiMessageVectorCtrl
```

List of all members.
<table>
<thead>
<tr>
<th>bool</th>
<th><strong>attach</strong> <em>(MessageVector item)</em></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Push a line onto the back of the list.</td>
</tr>
<tr>
<td>void</td>
<td><strong>detach</strong> ()</td>
</tr>
<tr>
<td></td>
<td>Stop listing messages from the MessageVector previously attached to, if any.</td>
</tr>
</tbody>
</table>
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>allowedMatches</td>
</tr>
<tr>
<td>int</td>
<td>lineContinuedIndex</td>
</tr>
<tr>
<td>int</td>
<td>lineSpacing</td>
</tr>
<tr>
<td>ColorI</td>
<td>matchColor</td>
</tr>
<tr>
<td>int</td>
<td>maxColorIndex</td>
</tr>
</tbody>
</table>
Detailed Description

A chat HUD control that displays messages from a MessageVector.

This renders messages from a MessageVector; the important thing here is that the MessageVector holds all the messages we care about, while we can destroy and create these GUI controls as needed.

Example:

```plaintext
// Declare ChatHud, which is what will display messages from MessageVector.
new GuiMessageVectorCtrl(ChatHud) {
    profile = "ChatHudMessageProfile";
    horizSizing = "width";
    vertSizing = "height";
    position = "1 1";
    extent = "252 16";
    minExtent = "8 8";
    visible = "1";
    helpTag = "0";
    lineSpacing = "0";
    lineContinuedIndex = "10";
    matchColor = "0 0 255 255";
    maxColorIndex = "5";
};

// All messages are stored in this HudMessageVector.
// MainChatHud only displays the contents of this vector.
new MessageVector(HudMessageVector);

// Attach the MessageVector to the chat control.
chatHud.attach(HudMessageVector);
```

See also:
MessageVector for more details on how this is used
### Member Function Documentation

#### `bool GuiMessageVectorCtrl::attach(MessageVector item)`

Push a line onto the back of the list.

**Parameters:**

- `item` The GUI element being pushed into the control

**Example:**

```cpp
// All messages are stored in this HudMessageVector
// MainChatHud only displays the contents of this vector
new MessageVector(HudMessageVector);

// Attach the MessageVector to the chat control
chatHud.attach(HudMessageVector);
```

**Returns:**

- Value

#### `void GuiMessageVectorCtrl::detach()`

Stop listing messages from the `MessageVector` previously attached to, if any.

**Detailed description**

**Parameters:**

- `param` Description

**Example:**

```cpp
// Deatch the MessageVector from HudMessageVector
// HudMessageVector will no longer render
```
chatHud.detach();
## Member Data Documentation

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>GuiMessageVectorCtrl::allowedMatches[16]</td>
</tr>
<tr>
<td>int</td>
<td>GuiMessageVectorCtrl::lineContinuedIndex</td>
</tr>
<tr>
<td>int</td>
<td>GuiMessageVectorCtrl::lineSpacing</td>
</tr>
<tr>
<td>ColorI</td>
<td>GuiMessageVectorCtrl::matchColor</td>
</tr>
<tr>
<td>int</td>
<td>GuiMessageVectorCtrl::maxColorIndex</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
GuiMLTextCtrl Class Reference

[Core Controls]

A text control that uses the Gui Markup Language ('ML') tags to dynamically change the text. More...

Inheritance diagram for GuiMLTextCtrl:

```
+-------------------+        +-------------------+
| SimObject         |        | SimSet            |
|                   |        |                   |
| +-------------------+        +-------------------+
| | SimGroup         |        | GuiControl        |
| +-------------------+        +-------------------+
| | GuiMLTextCtrl    |        | GuiMLTextEditCtrl |
| +-------------------+        +-------------------+
|                     |        | [legend]          |
```

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>addText</code> (string <code>text</code>, bool <code>reformat</code>)</td>
<td>Appends the text in the control with additional text. Also</td>
</tr>
<tr>
<td><code>forceReflow</code> ()</td>
<td>Forces the text control to reflow the text after new text is added, possibly resizing the control.</td>
</tr>
<tr>
<td><code>getText</code> ()</td>
<td>Returns the text from the control, including TorqueML characters.</td>
</tr>
<tr>
<td><code>scrollToBottom</code> ()</td>
<td>Scroll to the bottom of the text.</td>
</tr>
<tr>
<td><code>scrollToTag</code> (int <code>tagID</code>)</td>
<td>Scroll down to a specified tag.</td>
</tr>
<tr>
<td><code>scrollToTop</code> (int <code>param1</code>, int <code>param2</code>)</td>
<td>Scroll to the top of the text.</td>
</tr>
<tr>
<td><code>setAlpha</code> (float <code>alphaVal</code>)</td>
<td>Sets the alpha value of the control.</td>
</tr>
<tr>
<td><code>setCursorPosition</code> (int <code>newPos</code>)</td>
<td>Change the text cursor's position to a new defined offset within the text in the control.</td>
</tr>
<tr>
<td><code>setText</code> (string <code>text</code>)</td>
<td>Set the text contained in the control.</td>
</tr>
</tbody>
</table>

## Callbacks

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>onResize</code> (string <code>width</code>, string <code>maxY</code>)</td>
<td>Called whenever the control size changes.</td>
</tr>
<tr>
<td><code>onURL</code> (string <code>url</code>)</td>
<td>Called whenever a URL was clicked on within the control.</td>
</tr>
</tbody>
</table>
# Public Attributes

## Text

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>allowColorChars</td>
<td>If true, the control will allow characters to have unique colors.</td>
</tr>
<tr>
<td>SFXTrack</td>
<td>deniedSound</td>
<td>If the text will not fit in the control, the deniedSound is played.</td>
</tr>
<tr>
<td>int</td>
<td>lineSpacing</td>
<td>The number of blank pixels to place between each line.</td>
</tr>
<tr>
<td>int</td>
<td>maxChars</td>
<td>Maximum number of characters that the control will display.</td>
</tr>
<tr>
<td>caseString</td>
<td>text</td>
<td>Text to display in this control.</td>
</tr>
<tr>
<td>bool</td>
<td>useURLMouseCursor</td>
<td>If true, the mouse cursor will turn into a hand cursor while over a link in the text.</td>
</tr>
</tbody>
</table>
Detailed Description

A text control that uses the Gui Markup Language ('ML') tags to dynamically change the text.

Example of dynamic changes include colors, styles, and/or hyperlinks. These changes can occur without having to use separate text controls with separate text profiles.

Example:

```java
new GuiMLTextCtrl(CenterPrintText)
{
    lineSpacing = "2";
    allowColorChars = "0";
    maxChars = "-1";
    deniedSound = "DeniedSoundProfile";
    text = "The Text for This Control.";
    useURLMouseCursor = "true";
    //Properties not specific to this cont
}
```

See also:

GuiControl
void GuiMLTextCtrl::addText(string text, bool reformat)

Appends the text in the control with additional text. Also .

Parameters:
  text  New text to append to the existing text.
  reformat  If true, the control will also be visually reset.

Example:

```
// Define new text to add
%text = "New Text to Add";

// Set reformat boolean
%reformat = "true";

// Inform the control to add the new text
%thisGuiMLTextCtrl.addText(%text,%reformat)
```

See also:
  GuiControl

void GuiMLTextCtrl::forceReflow()

Forces the text control to reflow the text after new text is added, possibly resizing the control.

Example:

```
// Define new text to add
%newText = "BACON!";
```
// Add the new text to the control
%thisGuiMLTextCtrl.addText(%newText);

// Inform the GuiMLTextCtrl object to force a reflow to ensure the added text fits properly.
%thisGuiMLTextCtrl.forceReflow();

See also:
GuiControl

string GuiMLTextCtrl::getText()

Returns the text from the control, including TorqueML characters.

Example:

// Get the text displayed in the control
%controlText = %thisGuiMLTextCtrl.getText();

Returns:
Text string displayed in the control, including any TorqueML characters.

See also:
GuiControl

void GuiMLTextCtrl::onResize(string width, string maxY)

Called whenever the control size changes.

Parameters:

width  The new width value for the control
maxY The current maximum allowed Y value for the control

Example:

```cpp
// Control size changed, causing the callback
GuiMLTextCtrl::onResize(%this,%width,%maxY)
{
    // Code to call when the control size changed
}
```

See also:
GuiControl

void GuiMLTextCtrl::onURL(const string url)

Called whenever a URL was clicked on within the control.

Parameters:

url The URL address that was clicked on.

Example:

```cpp
// A URL address was clicked on in the control
GuiMLTextCtrl::onURL(%this,%url)
{
    // Code to run whenever a URL was clicked
}
```

See also:
GuiControl

void GuiMLTextCtrl::scrollToBottom()

Scroll to the bottom of the text.
Example:

```cpp
// Inform GuiMLTextCtrl object to scroll to its bottom
%thisGuiMLTextCtrl.scrollToBottom();
```

See also:
GuiControl

```cpp
void GuiMLTextCtrl::scrollToTag(int tagID )
```

Scroll down to a specified tag.

Detailed description

**Parameters:**

- `tagID` TagID to scroll the control to

Example:

```cpp
// Define the TagID we want to scroll the control to
%tagId = "4";

// Inform the GuiMLTextCtrl to scroll to a specified tag
%thisGuiMLTextCtrl.scrollToTag(%tagId);
```

See also:
GuiControl

```cpp
void GuiMLTextCtrl::scrollToTop(int param1,
    int param2 )
```

Scroll to the top of the text.

Example:
Inform GuiMLTextCtrl object to scroll to its top:
%thisGuiMLTextCtrl.scrollToTop();

See also:
GuiControl

void GuiMLTextCtrl::setAlpha(float alphaVal)

Sets the alpha value of the control.

Parameters:

alphaVal n - 1.0 floating value for the alpha

Example:

// Define the alpha value
%alphaVal = "0.5";

// Inform the control to update its alpha
%thisGuiMLTextCtrl.setAlpha(%alphaVal);

See also:
GuiControl

bool GuiMLTextCtrl::setCursorPosition(int newPos)

Change the text cursor's position to a new defined offset within the text in the control.

Parameters:

newPos Offset to place cursor.

Example:
// Define cursor offset position
%position = "23";

// Inform the GuiMLTextCtrl object to move
%thisGuiMLTextCtrl.setCursorPosition(%position);

Returns:
Returns true if the cursor position moved, or false if the position was not changed.

See also:
GuiControl

void GuiMLTextCtrl::setText(string text )

Set the text contained in the control.

Parameters:

  text  The text to display in the control.

Example:

// Define the text to display
%text = "Nifty Control Text";

// Set the text displayed within the control
%thisGuiMLTextCtrl.setText(%text);

See also:
GuiControl
**Member Data Documentation**

<table>
<thead>
<tr>
<th>Type</th>
<th>Member</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>GuiMLTextCtrl::allowColorChars</td>
</tr>
<tr>
<td></td>
<td>If true, the control will allow characters to have unique colors.</td>
</tr>
<tr>
<td>SFXTrack</td>
<td>GuiMLTextCtrl::deniedSound</td>
</tr>
<tr>
<td></td>
<td>If the text will not fit in the control, the deniedSound is played.</td>
</tr>
<tr>
<td>int</td>
<td>GuiMLTextCtrl::lineSpacing</td>
</tr>
<tr>
<td></td>
<td>The number of blank pixels to place between each line.</td>
</tr>
<tr>
<td>int</td>
<td>GuiMLTextCtrl::maxChars</td>
</tr>
<tr>
<td></td>
<td>Maximum number of characters that the control will display.</td>
</tr>
<tr>
<td>caseString</td>
<td>GuiMLTextCtrl::text</td>
</tr>
<tr>
<td></td>
<td>Text to display in this control.</td>
</tr>
<tr>
<td>bool</td>
<td>GuiMLTextCtrl::useURLMouseCursor</td>
</tr>
<tr>
<td></td>
<td>If true, the mouse cursor will turn into a hand cursor while over a link in the text.</td>
</tr>
<tr>
<td></td>
<td>This is dependant on the markup language used by the GuiMLTextCtrl</td>
</tr>
</tbody>
</table>
Copyright © GarageGames, LLC. All Rights Reserved.
GuiMLTextEditCtrl Class Reference
[General Controls]

A text entry control that accepts the Gui Markup Language ('ML') tags and multiple lines. More...

Inheritance diagram for GuiMLTextEditCtrl:

List of all members.
### Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td><code>escapeCommand</code></td>
<td>Script function to run whenever the 'escape' key is pressed when this control is in focus.</td>
</tr>
</tbody>
</table>
**Detailed Description**

A text entry control that accepts the Gui Markup Language ('ML') tags and multiple lines.

**Example:**

```java
new GuiMLTextEditCtrl()
{
    lineSpacing = "2";
    allowColorChars = "0";
    maxChars = "-1";
    deniedSound = "DeniedSoundProfile";
    text = "";
    escapeCommand = "onEscapeScriptFunction();"
    //Properties not specific to this control have been omitted from this example.
};
```

**See also:**

GuiMLTextCtrl
GuiControl
Member Data Documentation

| string GuiMLTextEditCtrl::escapeCommand |

Script function to run whenever the 'escape' key is pressed when this control is in focus.

Copyright © GarageGames, LLC. All Rights Reserved.
GuiMouseEventCtrl Class Reference
[Core Controls]

Used to overlaps a 'hot region' where you want to catch inputs with and have specific events occur based on individual callbacks. More...

Inheritance diagram for GuiMouseEventCtrl:

List of all members.
### Public Member Functions

#### Callbacks

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td><code>onMouseDown</code></td>
<td>Callback that occurs whenever the mouse is pressed down while in this control.</td>
</tr>
<tr>
<td>void</td>
<td><code>onMouseDragged</code></td>
<td>Callback that occurs whenever the mouse is dragged while in this control.</td>
</tr>
<tr>
<td>void</td>
<td><code>onMouseEnter</code></td>
<td>Callback that occurs whenever the mouse enters this control.</td>
</tr>
<tr>
<td>void</td>
<td><code>onMouseLeave</code></td>
<td>Callback that occurs whenever the mouse leaves this control.</td>
</tr>
<tr>
<td>void</td>
<td><code>onMouseMove</code></td>
<td>Callback that occurs whenever the mouse is moved (without dragging) while in this control.</td>
</tr>
<tr>
<td>void</td>
<td><code>onMouseUp</code></td>
<td>Callback that occurs whenever the mouse is released while in this control.</td>
</tr>
<tr>
<td>void</td>
<td><code>onRightMouseDown</code></td>
<td>Callback that occurs whenever the right mouse button is pressed while in this control.</td>
</tr>
<tr>
<td>void</td>
<td><code>onRightMouseDragged</code></td>
<td></td>
</tr>
</tbody>
</table>
Callback that occurs whenever the mouse is dragged in this control while the right mouse button is pressed.

<table>
<thead>
<tr>
<th>void onRightMouseUp (U8 modifier, Point2I mousePoint, U8 mouseClickCount)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Callback that occurs whenever the right mouse button is released while in this control.</td>
</tr>
</tbody>
</table>
Public Attributes

Input

bool lockMouse

Whether the control should lock the mouse between up and down button events.
Detailed Description

Used to overlaps a 'hot region' where you want to catch inputs with and have specific events occur based on individual callbacks.

Mouse event callbacks supported by this control are: onMouseUp, onMouseDown, onMouseMove, onMouseDragged, onMouseEnter, onMouseLeave, onRightMouseDown, onRightMouseUp and onRightMouseDragged.

Example:

```java
new GuiMouseEventCtrl() {
    lockMouse = "0";
    //Properties not specific to this control
};
```

See also:

GuiControl
void GuiMouseEventCtrl::onMouseDown (U8 modifier, Point2I mousePoint, U8 mouseClickCount)

Callback that occurs whenever the mouse is pressed down while in this control.

Parameters:

modifier Key that was pressed during this callback. Values are:

$EventModifier::RSHIFT
$EventModifier::SHIFT
$EventModifier::LCTRL
$EventModifier::RCTRL
$EventModifier::CTRL
$EventModifier::CTRL
$EventModifier::RALT
$EventModifier::ALT

Parameters:

mousePoint X/Y location of the mouse point

mouseClickCount How many mouse clicks have occurred for this event

Example:

// Mouse was pressed down in this control,
GuiMouseEventCtrl::onMouseDown(%this,%modifier,%mousePoint,%mouseClickCount) {
   // Code to call when a mouse event occurs.
}

See also:
   GuiControl

void GuiMouseEventCtrl::onMouseDragged(U8 modifier, Point2I mousePoint, U8 mouseClickedCount) {

Callback that occurs whenever the mouse is dragged while in this control.

Parameters:
   modifier Key that was pressed during this callback. Values are:

   $EventModifier::RSHIFT
   $EventModifier::SHIFT
   $EventModifier::LCTRL
   $EventModifier::RCTRL
   $EventModifier::CTRL
   $EventModifier::CTRL
   $EventModifier::RALT
   $EventModifier::ALT

Parameters:
mousePoint  X/Y location of the mouse point
mouseClickCount How many mouse clicks have occurred for this event

Example:

```
// Mouse was dragged in this control, causing
GuiMouseEventCtrl::onMouseDragged(%this,%modifier,%mousePoint,%mouseClickCount)
{
    // Code to call when a mouse event occurs.
}
```

See also:

GuiControl

```
void GuiMouseEventCtrl::onMouseEnter(U8 modifier,
Point2I mousePoint,
U8 mouseClickCount)
```

Callback that occurs whenever the mouse enters this control.

**Parameters:**

*modifier* Key that was pressed during this callback. Values are:

$EventModifier::RSHIFT
$EventModifier::SHIFT
$EventModifier::LCTRL
$EventModifier::RCTRL
$EventModifier::CTRL
$EventModifier::CTRL
$EventModifier::RALT
$EventModifier::ALT

**Parameters:**

- `mousePoint`  
  X/Y location of the mouse point
- `mouseClickCount`  
  How many mouse clicks have occurred for this event

**Example:**

```
// Mouse entered this control, causing the
GuiMouseEventCtrl::onMouseEnter(%this,%modifier,%mousePoint,%mouseClickCount)
{
    // Code to call when a mouse event occurs
}
```

**See also:**

`GuiControl`

```cpp
void GuiMouseEventCtrl::onMouseLeave(U8 modifier, Point2I mousePoint, U8 mouseClickCount)
```

Callback that occurs whenever the mouse leaves this control.

**Parameters:**

- `modifier`  
  Key that was pressed during this callback. Values are:

  - `$EventModifier::RSHIFT`
  - `$EventModifier::SHIFT`
  - `$EventModifier::LCTRL`
$EventModifier::RCTRL
$EventModifier::CTRL
$EventModifier::CTRL
$EventModifier::RALT
$EventModifier::ALT

**Parameters:**

- `mousePoint` X/Y location of the mouse point
- `mouseClickCount` How many mouse clicks have occurred for this event

**Example:**

```cpp
// Mouse left this control, causing the callback
GuiMouseEventCtrl::onMouseLeave(%this,%modifier,%mousePoint,%mouseClickCount)
{
    // Code to call when a mouse event occurs.
}
```

**See also:**

GuiControl

```cpp
void GuiMouseEventCtrl::onMouseMove(U8 modifier,
Point2I mousePoint,
U8 mouseClickCount)
```

Callback that occurs whenever the mouse is moved (without dragging) while in this control.

**Parameters:**

Key that was pressed during this callback. Values
modifier are:

$EventModifier::RSHIFT
$EventModifier::SHIFT
$EventModifier::LCTRL
$EventModifier::RCTRL
$EventModifier::CTRL
$EventModifier::CTRL
$EventModifier::RALT
$EventModifier::ALT

Parameters:

- **mousePoint**  X/Y location of the mouse point
- **mouseClickCount**  How many mouse clicks have occurred for this event

Example:

```cpp
// Mouse was moved in this control, causingGuiMouseEventCtrl::onMouseMove(%this,%modifier,%mousePoint,%mouseClickCount)
{
    // Code to call when a mouse event occurs
}
```

See also:

GuiControl

```cpp
void GuiMouseEventCtrl::onMouseUp (U8 modifier, Point2I mousePoint, U8 mouseClickCount)
```
Callback that occurs whenever the mouse is released while in this control.

**Parameters:**

- **modifier**
  - Key that was pressed during this callback. Values are:
    - `$EventModifier::RSHIFT`
    - `$EventModifier::SHIFT`
    - `$EventModifier::LCTRL`
    - `$EventModifier::RCTRL`
    - `$EventModifier::CTRL`
    - `$EventModifier::RALT`
    - `$EventModifier::ALT`

**Parameters:**

- **mousePoint**
  - X/Y location of the mouse point

- **mouseClickCount**
  - How many mouse clicks have occurred for this event

**Example:**

```c++
// Mouse was released in this control, call
GuiMouseEventCtrl::onMouseUp(%this,%modifier:
{
    // Code to call when a mouse event occurs
}
```
See also:

GuiControl

```c
void GuiMouseEventCtrl::onRightMouseDown(U8 modifier,
Point2I mousePoint,
U8 mouseClickCount)
```

Callback that occurs whenever the right mouse button is pressed while in this control.

**Parameters:**

- **modifier**
  Key that was pressed during this callback. Values are:
  - $EventModifier::RSHIFT
  - $EventModifier::SHIFT
  - $EventModifier::LCTRL
  - $EventModifier::RCTRL
  - $EventModifier::CTRL
  - $EventModifier::CTRL
  - $EventModifier::RALT
  - $EventModifier::ALT

- **mousePoint**
  X/Y location of the mouse point

- **mouseClickCount**
  How many mouse clicks have occurred for this event

**Example:**
// Right mouse button was pressed in this control, causing the callback
GuiMouseEventCtrl::onRightMouseDown(%this,
{
    // Code to call when a mouse event occurs.
}

See also:
GuiControl

void GuiMouseEventCtrl::onRightMouseDragged(U8 modifier,
    Point2I mousePoint,
    U8 mouseClicked
)

Callback that occurs whenever the mouse is dragged in this control while the right mouse button is pressed.

Parameters:

    modifier Key that was pressed during this callback. Values are:

$EventModifier::RSHIFT
$EventModifier::SHIFT
$EventModifier::LCTRL
$EventModifier::RCTRL
$EventModifier::CTRL
$EventModifier::CTRL
$EventModifier::RALT
$EventModifier::ALT
Parameters:

- `mousePoint`  
  X/Y location of the mouse point

- `mouseClickCount`  
  How many mouse clicks have occurred for this event

Example:

```c
// Right mouse button was dragged in this control,
// calling the callback:
GuiMouseEventCtrl::onRightMouseDragged()
{
    // Code to call when a mouse event occurs.
}
```

See also:

- `GuiControl`

```c
void GuiMouseEventCtrl::onRightMouseUp(U8 modifier, Point2I mousePoint, U8 mouseClickCount)
```

Callback that occurs whenever the right mouse button is released while in this control.

Parameters:

- `modifier`  
  Key that was pressed during this callback. Values are:

  - `$EventModifier::RSHIFT`
  - `$EventModifier::SHIFT`
  - `$EventModifier::LCTRL`
  - `$EventModifier::RCTRL`
Parameters:

- `mousePoint` X/Y location of the mouse point
- `mouseClickCount` How many mouse clicks have occurred for this event

Example:

```c++
// Right mouse button was released in this control,
GuiMouseEventCtrl::onRightMouseUp(%this, %modifier, %mousePoint, %mouseClickCount)
{
    // Code to call when a mouse event occurs
}
```

See also:

- `GuiControl`
Member Data Documentation

**bool GuiMouseEventCtrl::lockMouse**

Whether the control should lock the mouse between up and down button events.
GuiObjectView Class Reference
[3D Controls]

GUI control which displays a 3D model. More...

Inheritance diagram for GuiObjectView:

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>getCameraSpeed()</code></td>
<td>Return the current multiplier for camera zooming and rotation.</td>
</tr>
<tr>
<td>string</td>
<td><code>getModel()</code></td>
<td>Return the model displayed in this view.</td>
</tr>
<tr>
<td>string</td>
<td><code>getMountedModel()</code></td>
<td>Return the name of the mounted model.</td>
</tr>
<tr>
<td>string</td>
<td><code>getMountSkin(int param1, int param2)</code></td>
<td>Return the name of skin used on the mounted model.</td>
</tr>
<tr>
<td>float</td>
<td><code>getOrbitDistance()</code></td>
<td>Return the current distance at which the camera orbits the object.</td>
</tr>
<tr>
<td>string</td>
<td><code>getSkin()</code></td>
<td>Return the name of skin used on the primary model.</td>
</tr>
<tr>
<td>void</td>
<td><code>setCameraSpeed(float factor)</code></td>
<td>Sets the multiplier for the camera rotation and zoom speed.</td>
</tr>
<tr>
<td>void</td>
<td><code>setLightAmbient(ColorF color)</code></td>
<td>Set the light ambient color on the sun object used to render the model.</td>
</tr>
<tr>
<td>void</td>
<td><code>setLightColor(ColorF color)</code></td>
<td>Set the light color on the sun object used to render the model.</td>
</tr>
<tr>
<td>void</td>
<td><code>setLightDirection(Point3F direction)</code></td>
<td>Set the light direction from which to light the model.</td>
</tr>
<tr>
<td>void</td>
<td><code>setModel(string shapeName)</code></td>
<td>Sets the model to be displayed in this control.</td>
</tr>
<tr>
<td>void</td>
<td><code>setMount(string shapeName, string mountNodeIndexOrName)</code></td>
<td>Mounts the given model to the specified mount point of the</td>
</tr>
</tbody>
</table>
primary model displayed in this control.

<table>
<thead>
<tr>
<th>Method</th>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void setMountedModel (string shapeName)</code></td>
<td></td>
<td>Sets the model to be mounted on the primary model.</td>
</tr>
<tr>
<td><code>void setMountSkin (string skinName)</code></td>
<td></td>
<td>Sets the skin to use on the mounted model.</td>
</tr>
<tr>
<td><code>void setOrbitDistance (float distance)</code></td>
<td></td>
<td>Sets the distance at which the camera orbits the object. Clamped to the acceptable range defined in the class by min and max orbit distances.</td>
</tr>
<tr>
<td><code>void setSeq (string indexOrName)</code></td>
<td></td>
<td>Sets the animation to play for the viewed object.</td>
</tr>
<tr>
<td><code>void setSkin (string skinName)</code></td>
<td></td>
<td>Sets the skin to use on the model being displayed.</td>
</tr>
</tbody>
</table>

**Callbacks**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void onMouseEnter ()</code></td>
<td>Called whenever the mouse enters the control.</td>
</tr>
<tr>
<td><code>void onMouseLeave ()</code></td>
<td>Called whenever the mouse leaves the control.</td>
</tr>
</tbody>
</table>
### Public Attributes

#### Animation

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td><code>animSequence</code></td>
<td>The animation sequence to play on the model.</td>
</tr>
</tbody>
</table>

#### Camera

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>cameraSpeed</code></td>
<td>Multiplier for mouse camera operations.</td>
</tr>
<tr>
<td>float</td>
<td><code>maxOrbitDistance</code></td>
<td>Minimum distance below which the camera will not zoom in further.</td>
</tr>
<tr>
<td>float</td>
<td><code>minOrbitDistance</code></td>
<td>Maximum distance to which the camera can be zoomed out.</td>
</tr>
<tr>
<td>float</td>
<td><code>orbitDistance</code></td>
<td>Distance from which to render the model.</td>
</tr>
</tbody>
</table>

#### Lighting

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorF</td>
<td><code>lightAmbient</code></td>
<td>Ambient color of the sunlight used to render the model.</td>
</tr>
<tr>
<td>ColorF</td>
<td><code>lightColor</code></td>
<td>Diffuse color of the sunlight used to render the model.</td>
</tr>
<tr>
<td>Point3F</td>
<td><code>lightDirection</code></td>
<td>Direction from which the model is illuminated.</td>
</tr>
</tbody>
</table>

#### Mounting

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td><code>mountedNode</code></td>
<td>Name of node on primary model to which to mount the</td>
</tr>
</tbody>
</table>
secondary shape.

<table>
<thead>
<tr>
<th>filename</th>
<th>mountedShapeFile</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Optional shape file to mount on the primary model (e.g. weapon).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>mountedSkin</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Skin name used on mounted shape file.</td>
</tr>
</tbody>
</table>

**Model**

<table>
<thead>
<tr>
<th>filename</th>
<th>shapeFile</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The object model shape file to show in the view.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>skin</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The skin to use on the object model.</td>
</tr>
</tbody>
</table>
Detailed Description

GUI control which displays a 3D model.

Model displayed in the control can have other objects mounted onto it, and the light settings can be adjusted.

Example:

```java
new GuiObjectView(ObjectPreview) {
    shapeFile = "art/shapes/items/kit/healthkit.dts";
    mountedNode = "mount0";
    lightColor = "1 1 1 1";
    lightAmbient = "0.5 0.5 0.5 1";
    lightDirection = "0 0.707 -0.707";
    orbitDiststance = "2";
    minOrbitDiststance = "0.917688";
    maxOrbitDiststance = "5";
    cameraSpeed = "0.01";
    cameraZRot = "0";
    forceFOV = "0";
    reflectPriority = "0";
};
```

See also:

GuiControl
Member Function Documentation

float GuiObjectView::getCameraSpeed()

Return the current multiplier for camera zooming and rotation.

Example:

    // Request the current camera zooming and rotation multiplier value
    %multiplier = %thisGuiObjectView.getCameraSpeed();

Returns:
Camera zooming / rotation multiplier value.

See also:
GuiControl

string GuiObjectView::getModel()

Return the model displayed in this view.

Example:

    // Request the displayed model name from the GuiObjectView object
    %modelName = %thisGuiObjectView.getModel();

Returns:
Name of the displayed model.

See also:
GuiControl

string GuiObjectView::getMountedModel()
Return the name of the mounted model.

**Example:**

```cpp
// Request the name of the mounted model
%mountedModelName = %thisGuiObjectView.getMountedModelName();
```

**Returns:**

Name of the mounted model.

**See also:**

GuiControl

---

string GuiObjectView::getMountSkin(int param1,
                                    int param2
                                   )

Return the name of skin used on the mounted model.

**Example:**

```cpp
// Request the skin name from the model mounted on the main model in the control
%mountModelSkin = %thisGuiObjectView.getMountSkin();
```

**Returns:**

Name of the skin used on the mounted model.

**See also:**

GuiControl

---

float GuiObjectView::getOrbitDistance()

Return the current distance at which the camera orbits the object.
Example:

```c++
// Request the current orbit distance
%orbitDistance = %thisGuiObjectView.getOrbitDistance();
```

**Returns:**
The distance at which the camera orbits the object.

**See also:**
GuiControl

```c++
string GuiObjectView::getSkin( )
```

**Returns:**
Name of the skin used on the primary model.

**See also:**
GuiControl

```c++
void GuiObjectView::onMouseEnter( )
```

**Called whenever the mouse enters the control.**

**Example:**

```c++
// The mouse has entered the control, causing
GuiObjectView::onMouseEnter(%this)
```
// Code to run when the mouse enters
}

See also:
GuiControl

void GuiObjectView::onMouseLeave()

Called whenever the mouse leaves the control.

Example:

// The mouse has left the control, causing
GuiObjectView::onMouseLeave(%this)
{
    // Code to run when the mouse leaves
}

See also:
GuiControl

void GuiObjectView::setCameraSpeed(float factor)

Sets the multiplier for the camera rotation and zoom speed.

Parameters:

factor Multiplier for camera rotation and zoom speed.

Example:

// Set the factor value
%factor = "0.75";

// Inform the GuiObjectView object to set
See also:
GuiControl

void GuiObjectView::setLightAmbient(ColorF color )

Set the light ambient color on the sun object used to render the model.

Parameters:

color  Ambient color of sunlight.

Example:

```cpp
// Define the sun ambient color value
%color = "1.0 0.4 0.6";

// Inform the GuiObjectView object to set
%thisGuiObjectView.setLightAmbient(%color);
```

See also:
GuiControl

void GuiObjectView::setLightColor(ColorF color )

Set the light color on the sun object used to render the model.

Parameters:

color  Color of sunlight.

Example:

```cpp
// Set the color value for the sun
```
\%color = "1.0 0.4 0.5";

// Inform the GuiObjectView object to change the light color to the defined value
%thisGuiObjectView.setLightColor(%color);

See also:
GuiControl

void GuiObjectView::setLightDirection(Point3F direction)

Set the light direction from which to light the model.

Parameters:

\textit{direction} \quad \text{XYZ direction from which the light will shine on the model}

Example:

// Set the light direction
\%direction = "1.0 0.2 0.4"

// Inform the GuiObjectView object to change the light direction to the defined value
%thisGuiObjectView.setLightDirection(%direction);

See also:
GuiControl

void GuiObjectView::setModel(string shapeName)

Sets the model to be displayed in this control.

Parameters:

\textit{shapeName} \quad \text{Name of the model to display.}
Example:

```cpp
// Define the model we want to display
%shapeName = "gideon.dts";

// Tell the GuiObjectView object to display the defined model
%thisGuiObjectView.setModel(%shapeName);
```

See also:

`GuiControl`

```cpp
void GuiObjectView::setMount(string shapeName, string mountNodeIndexOrName)
```

Mounts the given model to the specified mount point of the primary model displayed in this control.

Detailed description

**Parameters:**

- `shapeName` Name of the model to mount.  
  Index or name of the mount point to be mounted to. If index, corresponds to "mountN" in your shape where N is the number passed here.

- `mountNodeIndexOrName`  

Example:

```cpp
// Set the shapeName to mount
%shapeName = "GideonGlasses.dts"

// Set the mount node of the primary model
%m MountNodeIndexOrName = "3";
// OR:
```
See also:

GuiControl

_Example:

```cpp
// Define the model name to mount
%modelToMount = "GideonGlasses.dts";

// Inform the GuiObjectView object to mount
%thisGuiObjectView.setMountedModel(%modelToMount);
```

See also:

GuiControl

void GuiObjectView::setMountSkin(string skinName )

Sets the skin to use on the mounted model.

Parameters:

- `skinName` Name of the skin to set on the model mounted to the main model in the control
Example:

```c
// Define the name of the skin
%skinName = "BronzeGlasses";

// Inform the GuiObjectView Control of the skin
%thisGuiObjectViewCtrl.setMountSkin(%skinName);
```

See also:

- GuiControl

---

```c
void GuiObjectView::setOrbitDistance(float distance )
```

Sets the distance at which the camera orbits the object. Clamped to the acceptable range defined in the class by min and max orbit distances.

Detailed description

**Parameters:**

- `distance` The distance to set the orbit to (will be clamped).

Example:

```c
// Define the orbit distance value
%orbitDistance = "1.5";

// Inform the GuiObjectView object to set the orbit distance
%thisGuiObjectView.setOrbitDistance(%orbitDistance);
```

See also:

- GuiControl

---

```c
void GuiObjectView::setSeq(string indexOrName )
```
Sets the animation to play for the viewed object.

Parameters:

\textit{indexOrName} The index or name of the animation to play.

Example:

```cpp
// Set the animation index value, or animation sequence name.
%indexVal = "3";
// OR:
%indexVal = "idle";
// Inform the GuiObjectView object to set
%thisGuiObjectView.setSeq(%indexVal);
```

See also:

GuiControl

---

\textbf{void GuiObjectView::setSkin(string skinName )}

Sets the skin to use on the model being displayed.

Parameters:

\textit{skinName} Name of the skin to use.

Example:

```cpp
// Define the skin we want to apply to the model.
%skinName = "disco_gideon";
// Inform the GuiObjectView control to update the skin
%thisGuiObjectView.setSkin(%skinName);
```

See also:

GuiControl
## Member Data Documentation

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Member Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>GuiObjectView::animSequence</td>
</tr>
<tr>
<td>The animation sequence to play on the model.</td>
<td></td>
</tr>
</tbody>
</table>

| float     | GuiObjectView::cameraSpeed |
| Multiplier for mouse camera operations. |

| ColorF    | GuiObjectView::lightAmbient |
| Ambient color of the sunlight used to render the model. |

| ColorF    | GuiObjectView::lightColor |
| Diffuse color of the sunlight used to render the model. |

| Point3F   | GuiObjectView::lightDirection |
| Direction from which the model is illuminated. |

| float     | GuiObjectView::maxOrbitDistance |
| Minimum distance below which the camera will not zoom in further. |

| float     | GuiObjectView::minOrbitDistance |
Maximum distance to which the camera can be zoomed out.

**string GuiObjectView::mountedNode**

Name of node on primary model to which to mount the secondary shape.

**filename GuiObjectView::mountedShapeFile**

Optional shape file to mount on the primary model (e.g. weapon).

**string GuiObjectView::mountedSkin**

Skin name used on mounted shape file.

**float GuiObjectView::orbitDistance**

Distance from which to render the model.

**filename GuiObjectView::shapeFile**

The object model shape file to show in the view.

**string GuiObjectView::skin**

The skin to use on the object model.
GuiPaneControl Class Reference
[Container Controls]

A collapsable pane control. More...

Inheritance diagram for GuiPaneControl:

List of all members.
Public Member Functions

void setCollapsed (bool collapse)

Collapse or un-collapse the control.
## Public Attributes

### Pane

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>barBehindText</td>
<td>Whether to draw the bitmapped pane bar behind the header text, too.</td>
</tr>
<tr>
<td>string</td>
<td>caption</td>
<td>Text label to display as the pane header.</td>
</tr>
<tr>
<td>string</td>
<td>captionID</td>
<td>String table text ID to use as caption string (overrides 'caption').</td>
</tr>
<tr>
<td>bool</td>
<td>collapsable</td>
<td>Whether the pane can be collapsed by clicking its header.</td>
</tr>
</tbody>
</table>
**Detailed Description**

A collapsable pane control.

This class wraps a single child control and displays a header with caption above it. If you click the header it will collapse or expand (if `collapsible` is enabled). The control resizes itself based on its collapsed/expanded size.

In the GUI editor, if you just want the header you can make `collapsible` false. The caption field lets you set the caption; it expects a bitmap (from the `GuiControlProfile`) that contains two images - the first is displayed when the control is expanded and the second is displayed when it is collapsed. The header is sized based on the first image.

**Example:**

```java
new GuiPaneControl()
{
    caption = "Example Pane";
    collapsable = "1";
    barBehindText = "1";
    //Properties not specific to this control
};
```
Member Function Documentation

void GuiPaneControl::setCollapsed(bool collapse)

Collapse or un-collapse the control.

**Parameters:**

*collapse*  True to collapse the control, false to un-collapse it
### Member Data Documentation

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool GuiPaneControl::barBehindText</td>
<td>Whether to draw the bitmapped pane bar behind the header text, too.</td>
</tr>
<tr>
<td>string GuiPaneControl::caption</td>
<td>Text label to display as the pane header.</td>
</tr>
<tr>
<td>string GuiPaneControl::captionID</td>
<td>String table text ID to use as caption string (overrides 'caption').</td>
</tr>
<tr>
<td>bool GuiPaneControl::collapsible</td>
<td>Whether the pane can be collapsed by clicking its header.</td>
</tr>
</tbody>
</table>
GuiPanel Class Reference
[Container Controls]

The GuiPanel panel is a container that when opaque will draw a left to right gradient using its profile fill and fill highlight colors. More...

Inheritance diagram for GuiPanel:

```
SimObject

SimSet

SimGroup

GuiControl

GuiContainer

GuiPanel
```

List of all members.
Detailed Description

The GuiPanel panel is a container that when opaque will draw a left to right gradient using its profile fill and fill highlight colors.

Example:

```java
// Mandatory GuiDefaultProfile
// Contains the fill color information required by GuiPanel
// Some values left out for sake of this example
new GuiControlProfile (GuiDefaultProfile) {
    // fill color
    opaque = false;
    fillColor = "242 241 240";
    fillColorHL = "228 228 235";
    fillColorSEL = "98 100 137";
    fillColorNA = "255 255 255 ";
};

new GuiPanel (TestPanel) {
    position = "45 33";
    extent = "342 379";
    minExtent = "16 16";
    horizSizing = "right";
    vertSizing = "bottom";
    profile = "GuiDefaultProfile"; // Color fill info is in this profile
    isContainer = "1";
};
```

See also:

GuiControlProfile
GuiPopUpMenuCtrl Class Reference  
[General Controls]

A control that allows to select a value from a drop-down list. More...

Inheritance diagram for GuiPopUpMenuCtrl:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void add(string name, int idNum, int scheme=0)</code></td>
<td>Adds a new entry with the specified name and ID.</td>
</tr>
<tr>
<td><code>void addScheme(int id, ColorI fontColor, ColorI fontColorHL, ColorI fontColorSEL)</code></td>
<td>Adds a new scheme with the specified color information.</td>
</tr>
<tr>
<td><code>void changeTextById(int id, string text)</code></td>
<td>Changes the text of the entry with the specified ID.</td>
</tr>
<tr>
<td><code>void clearEntry(S32 entry)</code></td>
<td>Clears the entry with the specified ID.</td>
</tr>
<tr>
<td><code>int findText(string text)</code></td>
<td>Returns the position of the first entry containing the specified text.</td>
</tr>
<tr>
<td><code>string getTextById(int id)</code></td>
<td>Gets the text of the entry with the specified ID.</td>
</tr>
<tr>
<td><code>void replaceText(bool doReplaceText)</code></td>
<td>Replaces the text of the entry with the specified ID.</td>
</tr>
<tr>
<td><code>void setEnumContent(string class, string enum)</code></td>
<td>Fills the popup with a classrep's field enumeration type info.</td>
</tr>
<tr>
<td><code>void setFirstSelected([scriptCallback=true])</code></td>
<td>Selects the first entry.</td>
</tr>
<tr>
<td><code>void setSelected(int id,[scriptCallback=true])</code></td>
<td>Selects the entry with the specified ID.</td>
</tr>
</tbody>
</table>
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filename</td>
<td>bitmap</td>
<td></td>
</tr>
<tr>
<td>Point2I</td>
<td>bitmapBounds</td>
<td></td>
</tr>
<tr>
<td>void</td>
<td>clear</td>
<td>Clear the popup list.</td>
</tr>
<tr>
<td>void</td>
<td>forceClose</td>
<td></td>
</tr>
<tr>
<td>void</td>
<td>forceOnAction</td>
<td></td>
</tr>
<tr>
<td>int</td>
<td>getSelected</td>
<td></td>
</tr>
<tr>
<td>string</td>
<td>getText</td>
<td></td>
</tr>
<tr>
<td>int</td>
<td>maxPopupHeight</td>
<td></td>
</tr>
<tr>
<td>bool</td>
<td>reverseTextList</td>
<td></td>
</tr>
<tr>
<td>bool</td>
<td>sbUsesNAColor</td>
<td></td>
</tr>
<tr>
<td>void</td>
<td>setNoneSelected</td>
<td></td>
</tr>
<tr>
<td>int</td>
<td>size</td>
<td>Get the size of the menu - the number of entries in it.</td>
</tr>
<tr>
<td>void</td>
<td>sort</td>
<td>Sort the list alphabetically.</td>
</tr>
<tr>
<td>void</td>
<td>sortID</td>
<td>Sort the list by ID.</td>
</tr>
</tbody>
</table>
Detailed Description

A control that allows to select a value from a drop-down list.

For a nearly identical GUI with additional features, use 
GuiPopUpMenuCtrlEx.

Example:

```java
new GuiPopUpMenuCtrl1()
{
    maxPopupHeight = "200";
    sbUsesNAColor = "0";
    reverseTextList = "0";
    bitmapBounds = "16 16";
    maxLength = "1024";
    position = "56 31";
    extent = "64 64";
    minExtent = "8 2";
    profile = "GuiPopUpMenuProfile";
    tooltipProfile = "GuiToolTipProfile";
};
```

Note:
This is definitely going to be deprecated soon.

See also:
GuiPopUpMenuCtrlEx for more features and better explanations.
### Member Function Documentation

<table>
<thead>
<tr>
<th>Function</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void GuiPopUpMenuCtrl::add(string name, int idNum, int scheme = 0)</code></td>
<td></td>
</tr>
<tr>
<td><code>void GuiPopUpMenuCtrl::addScheme(int id, ColorI fontColor, ColorI fontColorHL, ColorI fontColorSEL)</code></td>
<td></td>
</tr>
<tr>
<td><code>void GuiPopUpMenuCtrl::changeTextById(int id, string text)</code></td>
<td></td>
</tr>
<tr>
<td><code>void GuiPopUpMenuCtrl::clearEntry(S32 entry)</code></td>
<td></td>
</tr>
<tr>
<td><code>int GuiPopUpMenuCtrl::findText(string text)</code></td>
<td></td>
</tr>
<tr>
<td>Returns the position of the first entry containing the specified text.</td>
<td></td>
</tr>
<tr>
<td><code>string GuiPopUpMenuCtrl::getTextById(int id)</code></td>
<td></td>
</tr>
<tr>
<td><code>void GuiPopUpMenuCtrl::replaceText(bool doReplaceText)</code></td>
<td></td>
</tr>
<tr>
<td><code>void GuiPopUpMenuCtrl::setEnumContent(string class, string enum)</code></td>
<td></td>
</tr>
</tbody>
</table>
This fills the popup with a classrep's field enumeration type info.

More of a helper function than anything. If console access to the field list is added, at least for the enumerated types, then this should go away.

```cpp
void GuiPopUpMenuCtrl::setFirstSelected( )

void GuiPopUpMenuCtrl::setSelected(int id )
```
Member Data Documentation

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filename GuiPopUpMenuCtrl::bitmap</td>
<td></td>
</tr>
<tr>
<td>Point2I GuiPopUpMenuCtrl::bitmapBounds</td>
<td></td>
</tr>
<tr>
<td>void GuiPopUpMenuCtrl::clear</td>
<td>Clear the popup list.</td>
</tr>
<tr>
<td>void GuiPopUpMenuCtrl::forceClose</td>
<td></td>
</tr>
<tr>
<td>void GuiPopUpMenuCtrl::forceOnAction</td>
<td></td>
</tr>
<tr>
<td>int GuiPopUpMenuCtrl::getSelected</td>
<td></td>
</tr>
<tr>
<td>string GuiPopUpMenuCtrl::getText</td>
<td></td>
</tr>
<tr>
<td>int GuiPopUpMenuCtrl::maxPopupHeight</td>
<td></td>
</tr>
<tr>
<td>bool GuiPopUpMenuCtrl::reverseTextList</td>
<td></td>
</tr>
<tr>
<td>bool GuiPopUpMenuCtrl::sbUsesNAColor</td>
<td></td>
</tr>
<tr>
<td>void GuiPopUpMenuCtrl::setNoneSelected</td>
<td></td>
</tr>
<tr>
<td>int GuiPopUpMenuCtrl::size</td>
<td>Get the size of the menu - the number of entries in it.</td>
</tr>
</tbody>
</table>
void GuiPopUpMenuCtrl::sort

Sort the list alphabetically.

void GuiPopUpMenuCtrl::sortID

Sort the list by ID.
GuiPopUpMenuCtrlEx Class Reference
[General Controls]

A control that allows to select a value from a drop-down list. More...

Inheritance diagram for GuiPopUpMenuCtrlEx:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>void</th>
<th>add (string name, int idNum, int scheme=0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>add (string name, S32 idNum, S32 scheme=0)</td>
</tr>
<tr>
<td></td>
<td>Adds an entry to the list.</td>
</tr>
<tr>
<td>void</td>
<td>addCategory (string text)</td>
</tr>
<tr>
<td></td>
<td>Add a category to the list.</td>
</tr>
<tr>
<td>void</td>
<td>addScheme (int id, ColorI fontColor, ColorI fontColorHL, ColorI fontColorSEL)</td>
</tr>
<tr>
<td></td>
<td>Create a new scheme and add it to the list of choices for when a new text entry is added.</td>
</tr>
<tr>
<td>void</td>
<td>clear ()</td>
</tr>
<tr>
<td></td>
<td>Clear the popup list.</td>
</tr>
<tr>
<td>void</td>
<td>clearEntry (S32 entry)</td>
</tr>
<tr>
<td>int</td>
<td>findText (string text)</td>
</tr>
<tr>
<td></td>
<td>Returns the id of the first entry containing the specified text or -1 if not found.</td>
</tr>
<tr>
<td>void</td>
<td>forceClose ()</td>
</tr>
<tr>
<td></td>
<td>Manually force this control to collapse and close.</td>
</tr>
<tr>
<td>void</td>
<td>forceOnAction ()</td>
</tr>
<tr>
<td></td>
<td>Manually for the onAction function, which updates everything in this control.</td>
</tr>
<tr>
<td>int</td>
<td>getSelected ()</td>
</tr>
<tr>
<td></td>
<td>Get the current selection of the menu.</td>
</tr>
<tr>
<td>string</td>
<td>getText ()</td>
</tr>
<tr>
<td></td>
<td>Get the.</td>
</tr>
<tr>
<td>string</td>
<td>getTextById (int id)</td>
</tr>
<tr>
<td></td>
<td>Get the text of an entry based on an ID.</td>
</tr>
<tr>
<td>void</td>
<td>setNoneSelected (int param)</td>
</tr>
<tr>
<td></td>
<td>Clears selection in the menu.</td>
</tr>
<tr>
<td>setSelected (int id, bool scriptCallback=true)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------------------------------------------------</td>
</tr>
<tr>
<td><code>setSelected</code></td>
<td>(bool scriptCallback=true)</td>
</tr>
<tr>
<td><code>void setText (string text)</code></td>
<td>Set the current text to a specified value.</td>
</tr>
<tr>
<td><code>void sort ()</code></td>
<td>Sort the list alphabetically.</td>
</tr>
<tr>
<td><code>void sortID ()</code></td>
<td>Sort the list by ID.</td>
</tr>
</tbody>
</table>
# Public Attributes

<table>
<thead>
<tr>
<th>Attribute Type</th>
<th>Attribute Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filename</td>
<td>bitmap</td>
<td>File name of bitmap to use.</td>
</tr>
<tr>
<td>Point2I</td>
<td>bitmapBounds</td>
<td>Boundaries of bitmap displayed.</td>
</tr>
<tr>
<td>string</td>
<td>getColorById</td>
<td>Get color of an entry's box.</td>
</tr>
<tr>
<td>bool</td>
<td>hotTrackCallback</td>
<td>Whether to provide a 'onHotTrackItem' callback when a list item is hovered over.</td>
</tr>
<tr>
<td>int</td>
<td>maxPopupHeight</td>
<td>Length of menu when it extends.</td>
</tr>
<tr>
<td>void</td>
<td>replaceText</td>
<td>Flag that causes each new text addition to replace the current entry.</td>
</tr>
<tr>
<td>bool</td>
<td>reverseTextList</td>
<td>Reverses text list if popup extends up, instead of down.</td>
</tr>
<tr>
<td>bool</td>
<td>sbUsesNAColor</td>
<td>Deprecated.</td>
</tr>
<tr>
<td>void</td>
<td>setEnumContent</td>
<td>This fills the popup with a classrep's field enumeration type info.</td>
</tr>
<tr>
<td>int</td>
<td>size</td>
<td>Get the size of the menu.</td>
</tr>
</tbody>
</table>
**Detailed Description**

A control that allows to select a value from a drop-down list.

This is essentially a `GuiPopUpMenuCtrl`, but with quite a few more features.

**Example:**

```java
new GuiPopUpMenuCtrlEx()
{
    maxPopupHeight = "200";
    sbUsesNAColor = "0";
    reverseTextList = "0";
    bitmapBounds = "16 16";
    hotTrackCallback = "0";
    extent = "64 64";
    profile = "GuiDefaultProfile";
    tooltipProfile = "GuiToolTipProfile";
};
```

**See also:**

`GuiPopUpMenuCtrl`
## Member Function Documentation

### void GuiPopUpMenuCtrlEx::add (string name, int idNum, int scheme = 0 )

Adds an entry to the list.

**Parameters:**

- `name` String containing the name of the entry
- `idNum` Numerical value assigned to the name
- `scheme` Optional ID associated with a scheme for font coloring, highlight coloring, and selection coloring

### void GuiPopUpMenuCtrlEx::addCategory (string text )

Add a category to the list.

Acts as a separator between entries, allowing for sub-lists

**Parameters:**

- `text` Name of the new category

### void GuiPopUpMenuCtrlEx::addScheme (int id, ColorI fontColor, ColorI fontColorHL, ColorI fontColorSEL)
Create a new scheme and add it to the list of choices for when a new text entry is added.

**Parameters:**

- **id**  
  Numerical id associated with this scheme

- **fontColor**  
  The base text font color. Formatted as "Red Green Blue", each a numerical between 0 and 255.

- **fontColorHL**  
  Color of text when being highlighted. Formatted as "Red Green Blue", each a numerical between 0 and 255.

- **fontColorSel**  
  Color of text when being selected. Formatted as "Red Green Blue", each a numerical between 0 and 255.

---

**void GuiPopUpMenuCtrlEx::clear ()**

Clear the popup list.

Reimplemented from **SimSet**.

---

**void GuiPopUpMenuCtrlEx::clearEntry(S32 entry )**

---

**int GuiPopUpMenuCtrlEx::findText(string text )**

Returns the id of the first entry containing the specified text or -1 if not found.

**Parameters:**

- **text**  
  String value used for the query

**Returns:**
Numerical ID of entry containing the text.

```cpp
void GuiPopUpMenuCtrlEx::forceClose() {
    // Manually force this control to collapse and close.
}
```

```cpp
void GuiPopUpMenuCtrlEx::forceOnAction() {
    // Manually force the onAction function, which updates everything in this control.
}
```

```cpp
int GuiPopUpMenuCtrlEx::getSelected() {
    // Get the current selection of the menu.
    // Returns: Returns the ID of the currently selected entry
}
```

```cpp
string GuiPopUpMenuCtrlEx::getText() {
    // Get the.
    // Detailed description
    // Parameters: 
    //   param Description
    // Example:
    //     // Comment
code();
    // Returns:
}
```
Returns current text in string format

```cpp
string GuiPopUpMenuCtrlEx::getTextById (int id )
```

Get the text of an entry based on an ID.

**Parameters:**
- *id* The ID assigned to the entry being queried

**Returns:**
- String contained by the specified entry, NULL if empty or bad ID

```cpp
void GuiPopUpMenuCtrlEx::setNoneSelected (int param )
```

Clears selection in the menu.

```cpp
GuiPopUpMenuCtrlEx::setSelected (int id ,
    bool scriptCallback = true )
```

brief Manually set an entry as selected int his control

**Parameters:**
- *id* The ID of the entry to select
- *scriptCallback* Optional boolean that forces the script callback if true

```cpp
GuiPopUpMenuCtrlEx::setSelected (bool scriptCallback = true )
```

brief Manually set the selection to the first entry
Parameters:

scripCallback Optional boolean that forces the script callback if true

void GuiPopUpMenuCtrlEx::setText ( string text )

Set the current text to a specified value.

Parameters:

text String containing new text to set

Reimplemented from GuiTextCtrl.

void GuiPopUpMenuCtrlEx::sort ( )

Sort the list alphabetically.

void GuiPopUpMenuCtrlEx::sortID ( )

Sort the list by ID.
### Member Data Documentation

<table>
<thead>
<tr>
<th><strong>filename</strong> GuiPopupMenuCtrlEx::bitmap</th>
</tr>
</thead>
<tbody>
<tr>
<td>File name of bitmap to use.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Point2I GuiPopupMenuCtrlEx::bitmapBounds</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Boundaries of bitmap displayed.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>string GuiPopupMenuCtrlEx::getColorById</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Get color of an entry's box.</td>
</tr>
</tbody>
</table>

**Parameters:**

- **id** ID number of entry to query

**Returns:**

ColorI in the format of "Red Green Blue Alpha", each of with is a value between 0 - 255

<table>
<thead>
<tr>
<th><strong>bool GuiPopupMenuCtrlEx::hotTrackCallback</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Whether to provide a 'onHotTrackItem' callback when a list item is hovered over.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>int GuiPopupMenuCtrlEx::maxPopupHeight</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Length of menu when it extends.</td>
</tr>
</tbody>
</table>
**void GuiPopUpMenuCtrlEx::replaceText**

Flag that causes each new text addition to replace the current entry.

**Parameters:**

*True* to turn on replacing, false to disable it

**bool GuiPopUpMenuCtrlEx::reverseTextList**

Reverses text list if popup extends up, instead of down.

**bool GuiPopUpMenuCtrlEx::sbUsesNAColor**

Deprecated.

**void GuiPopUpMenuCtrlEx::setEnumContent**

This fills the popup with a classrep's field enumeration type info.

More of a helper function than anything. If console access to the field list is added, at least for the enumerated types, then this should go away.

**Parameters:**

*class*  Name of the class containing the enum  
*enum*  Name of the enum value to access

**int GuiPopUpMenuCtrlEx::size**

Get the size of the menu.
Returns:
   Number of entries in the menu
GuiProgressBitmapCtrl Class Reference
[Value Controls]

A horizontal progress bar rendered from a repeating image. More...

Inheritance diagram for GuiProgressBitmapCtrl:

```
SimObject
   |
   v
SimSet
   |
   v
SimGroup
   |
   v
GuiControl
   |
   v
GuiContainer
   |
   v
GuiTextCtrl
   |
 GuiProgressBitmapCtrl
```

[legend]

List of all members.
Public Member Functions

void **setBitmap** (string filename)
  Set the bitmap to use for rendering the progress bar.
Public Attributes

<table>
<thead>
<tr>
<th>filename</th>
<th>bitmap</th>
</tr>
</thead>
<tbody>
<tr>
<td>~Path to the bitmap file to use for rendering the progress bar.</td>
<td></td>
</tr>
</tbody>
</table>
**Detailed Description**

A horizontal progress bar rendered from a repeating image.

This class is used to give progress feedback to the user. Unlike `GuiProgressCtrl` which simply renders a filled rectangle, `GuiProgressBitmapCtrl` renders the bar using a bitmap.

This bitmap can either be simple, plain image which is then stretched into the current extents of the bar as it fills up or it can be a bitmap array with three entries. In the case of a bitmap array, the first entry in the array is used to render the left cap of the bar and the third entry in the array is used to render the right cap of the bar. The second entry is stretched in-between the two caps.

**Example:**

```plaintext
// This example shows one way to break down // and incrementally update a progress bar

new GuiProgressBitmapCtrl( Progress )
{
    bitmap = "core/art/gui/images/loading";
    extent = "300 50";
    position = "100 100";
};

// Put the control on the canvas.
%wrapper = new GuiControl();
%wrapper.addObject( Progress );
Canvas.pushDialog( %wrapper );

// Start the computation.
schedule( 1, 0, "phase1" );

function phase1()
```
{ Progress.setValue( 0 );

    // Perform some computation.  
    //...

    // Update progress.  
    Progress.setValue( 0.25 );

    // Schedule next phase.  Don't call directly so engine gets a change to run refresh.  
    schedule( 1, 0, "phase2" );
}

function phase2()
{
    // Perform some computation.  
    //...

    // Update progress.  
    Progress.setValue( 0.7 );

    // Schedule next phase.  Don't call directly so engine gets a change to run refresh.  
    schedule( 1, 0, "phase3" );
}

function phase3()
{  
    // Perform some computation.  
    //...

    // Update progress.  
    Progress.setValue( 0.9 );
}
// Schedule next phase. Don't call directly.
schedule( 1, 0, "phase4" );

function phase4()
{
    // Perform some computation.
    //...

    // Final update of progress.
    Progress.setValue( 1.0 );
}

See also:

GuiProgressCtrl
Member Function Documentation

void GuiProgressBitmapCtrl::setBitmap(string filename)

Set the bitmap to use for rendering the progress bar.

**Parameters:**

- *filename* ~Path to the bitmap file.

**Note:**

Directly assign to *bitmap* rather than using this method.

**See also:**

GuiProgressBitmapCtrl::setBitmap
Member Data Documentation

<table>
<thead>
<tr>
<th>filename</th>
<th>GuiProgressBitmapCtrl::bitmap</th>
</tr>
</thead>
</table>

~Path to the bitmap file to use for rendering the progress bar.

If the profile assigned to the control already has a bitmap assigned, this property need not be set in which case the bitmap from the profile is used.

Copyright © GarageGames, LLC. All Rights Reserved.
GuiProgressCtrl Class Reference
[Value Controls]

GUI Control which displays a horizontal bar which increases as the progress value of 0.0 - 1.0 increases. More...

Inheritance diagram for GuiProgressCtrl:

List of all members.
Detailed Description

GUI Control which displays a horizontal bar which increases as the progress value of 0.0 - 1.0 increases.

Example:

```java
new GuiProgressCtrl(JS_statusBar) {
    //Properties not specific to this control have been omitted from this example.
};

// Define the value to set the progress bar

// Set the value of the progress bar, from 0.0 - 1.0
%thisGuiProgressCtrl.setValue(%value);

// Get the value of the progress bar.
%progress = %thisGuiProgressCtrl.getValue();
```

See also:

GuiTextCtrl
GuiControl

Copyright © GarageGames, LLC. All Rights Reserved.
GuiRadioCtrl Class Reference
[Button Controls]

A button based around the radio concept. More...

Inheritance diagram for GuiRadioCtrl:

List of all members.
Detailed Description

A button based around the radio concept.

GuiRadioCtrl's functionality is based around GuiButtonBaseCtrl's ButtonTypeRadio type.

A button control with a radio box and a text label. This control is used in groups where multiple radio buttons present a range of options out of which one can be chosen. A radio button automatically signals its siblings when it is toggled on.

Example:

```cpp
// Create a GuiCheckBoxCtrl that calls randomFunction when clicked.
%radio = new GuiRadioCtrl()
{
  profile = "GuiRadioProfile";
};
```
GuiRolloutCtrl Class Reference
[Container Controls]

A container that shows a single child with an optional header bar that can be used to collapse and expand the rollout. More...

Inheritance diagram for GuiRolloutCtrl:

```
   SimObject
     
     SimSet
       
       SimGroup
         
         GuiControl
           
           GuiRolloutCtrl
```

List of all members.
# Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void collapse()</code></td>
<td>Collapse the rollout if it is currently expanded. This will make the rollout's child control invisible.</td>
</tr>
<tr>
<td><code>void expand()</code></td>
<td>Expand the rollout if it is currently collapsed. This will make the rollout's child control visible.</td>
</tr>
<tr>
<td><code>void instantCollapse()</code></td>
<td>Instantly collapse the rollout without animation. To smoothly slide the rollout to collapsed state, use <code>collapse()</code>.</td>
</tr>
<tr>
<td><code>void instantExpand()</code></td>
<td>Instantly expand the rollout without animation. To smoothly slide the rollout to expanded state, use <code>expand()</code>.</td>
</tr>
<tr>
<td><code>bool isExpanded()</code></td>
<td>Determine whether the rollout is currently expanded, i.e. whether the child control is visible.</td>
</tr>
<tr>
<td><code>void sizeToContents()</code></td>
<td>Resize the rollout to exactly fit around its child control. This can be used to manually trigger a recomputation of the rollout size.</td>
</tr>
<tr>
<td><code>void toggleCollapse()</code></td>
<td>Toggle the current collapse state of the rollout. If it is currently expanded, then collapse it. If it is currently collapsed, then expand it.</td>
</tr>
<tr>
<td><code>void toggleExpanded(bool instantly=false)</code></td>
<td>Toggle the current expansion state of the rollout. If it is currently expanded, then collapse it. If it is currently collapsed, then expand it.</td>
</tr>
</tbody>
</table>

## Callbacks

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void onCollapsed()</code></td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>void onExpanded()</td>
<td>Called when the rollout is expanded.</td>
</tr>
<tr>
<td>void onHeaderRightClick()</td>
<td>Called when the user right-clicks on the rollout's header. This is useful for implementing context menus for rollouts.</td>
</tr>
</tbody>
</table>
## Public Attributes

### Rollout

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>autoCollapseSiblings</td>
<td>Whether to automatically collapse sibling rollouts.</td>
</tr>
<tr>
<td>string</td>
<td>caption</td>
<td>Text label to display on the rollout header.</td>
</tr>
<tr>
<td>bool</td>
<td>clickCollapse</td>
<td>Whether the rollout can be collapsed by clicking its header.</td>
</tr>
<tr>
<td>int</td>
<td>defaultHeight</td>
<td>Default height of the client area. This is used when no child control has been added to the rollout.</td>
</tr>
<tr>
<td>bool</td>
<td>expanded</td>
<td>The current rollout expansion state.</td>
</tr>
<tr>
<td>bool</td>
<td>hideHeader</td>
<td>Whether to render the rollout header.</td>
</tr>
<tr>
<td>Rectl</td>
<td>margin</td>
<td>Margin to put around child control.</td>
</tr>
</tbody>
</table>
Detailed Description

A container that shows a single child with an optional header bar that can be used to collapse and expand the rollout.

A rollout is a container that can be collapsed and expanded using smooth animation. By default, rollouts will display a header with a caption along the top edge of the control which can be clicked by the user to toggle the collapse state of the rollout.

Rollouts will automatically size themselves to exactly fit around their child control. They will also automatically position their child control in their upper left corner below the header (if present).

Note:

GuiRolloutCtrls will only work correctly with a single child control. To put multiple controls in a rollout, put them in their own group using a new GuiControl which then can be put inside the rollout.
Member Function Documentation

void GuiRolloutCtrl::collapse(
)  
Collapse the rollout if it is currently expanded. This will make the rollout's child control invisible.

Note:  
The rollout will animate to collapsed state. To instantly collapse without animation, use instantCollapse().

void GuiRolloutCtrl::expand(
)  
Expand the rollout if it is currently collapsed. This will make the rollout's child control visible.

Note:  
The rollout will animate to expanded state. To instantly expand without animation, use instantExpand().

void GuiRolloutCtrl::instantCollapse(
)  
Instantly collapse the rollout without animation. To smoothly slide the rollout to collapsed state, use collapse().

void GuiRolloutCtrl::instantExpand(
)  
Instantly expand the rollout without animation. To smoothly slide the rollout to expanded state, use expand().

bool GuiRolloutCtrl::isExpanded(
)
Determine whether the rollout is currently expanded, i.e. whether the child control is visible.

**Returns:**

True if the rollout is expanded, false if not.

Reimplemented from *SimObject*.

```cpp
void GuiRolloutCtrl::onCollapsed()
```

Called when the rollout is collapsed.

```cpp
void GuiRolloutCtrl::onExpanded()
```

Called when the rollout is expanded.

```cpp
void GuiRolloutCtrl::onHeaderRightClick()
```

Called when the user right-clicks on the rollout's header. This is useful for implementing context menus for rollouts.

```cpp
void GuiRolloutCtrl::sizeToContents()
```

Resize the rollout to exactly fit around its child control. This can be used to manually trigger a recomputation of the rollout size.

```cpp
void GuiRolloutCtrl::toggleCollapse()
```

Toggle the current collapse state of the rollout. If it is currently expanded, then collapse it. If it is currently collapsed, then expand it.
void GuiRolloutCtrl::toggleExpanded(bool instantly = false)

Toggle the current expansion state of the rollout. If it is currently expanded, then collapse it. If it is currently collapsed, then expand it.

**Parameters:**

- **instant**
  If true, the rollout will toggle its state without *instant* animation. Otherwise, the rollout will smoothly slide into the opposite state.
Member Data Documentation

**bool GuiRolloutCtrl::autoCollapseSiblings**

Whether to automatically collapse sibling rollouts.

If this is true, the rollout will automatically collapse all sibling rollout controls when it is expanded. If this is false, the auto-collapse behavior can be triggered by CTRL (CMD on MAC) clicking the rollout header. CTRL/CMD clicking also works if this is false, in which case the auto-collapsing of sibling controls will be temporarily deactivated.

**string GuiRolloutCtrl::caption**

Text label to display on the rollout header.

**bool GuiRolloutCtrl::clickCollapse**

Whether the rollout can be collapsed by clicking its header.

**int GuiRolloutCtrl::defaultHeight**

Default height of the client area. This is used when no child control has been added to the rollout.

**bool GuiRolloutCtrl::expanded**

The current rollout expansion state.

**bool GuiRolloutCtrl::hideHeader**
Whether to render the rollout header.

**Note:**

If this is false, the user cannot toggle the rollout state with the mouse.

RectI GuiRolloutCtrl::margin

Margin to put around child control.
GuiScriptNotifyCtrl Class Reference

[Utility Controls]

A control which adds several reactions to other GUIs via callbacks.

More...

Inheritance diagram for GuiScriptNotifyCtrl:

```
SimObject

SimSet

SimGroup

GuiControl

GuiScriptNotifyCtrl
```

List of all members.
## Callbacks

<table>
<thead>
<tr>
<th>Type</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>onChildAdded</td>
<td>Enables/disables onChildAdded callback.</td>
</tr>
<tr>
<td>bool</td>
<td>onChildRemoved</td>
<td>Enables/disables onChildRemoved callback.</td>
</tr>
<tr>
<td>bool</td>
<td>onChildResized</td>
<td>Enables/disables onChildResized callback.</td>
</tr>
<tr>
<td>bool</td>
<td>onGainFirstResponder</td>
<td>Enables/disables onGainFirstResponder callback.</td>
</tr>
<tr>
<td>bool</td>
<td>onLoseFirstResponder</td>
<td>Enables/disables onLoseFirstResponder callback.</td>
</tr>
<tr>
<td>bool</td>
<td>onParentResized</td>
<td>Enables/disables onParentResized callback.</td>
</tr>
<tr>
<td>bool</td>
<td>onResize</td>
<td>Enables/disables onResize callback.</td>
</tr>
<tr>
<td>void</td>
<td>onChildAdded (SimObjectId ID, SimObjectId childID)</td>
<td>Called when a child is added to this GUI.</td>
</tr>
<tr>
<td>void</td>
<td>onChildRemoved (SimObjectId ID, SimObjectId childID)</td>
<td>Called when a child is removed from this GUI.</td>
</tr>
<tr>
<td>void</td>
<td>onChildResized (SimObjectId ID, SimObjectId childID)</td>
<td>Called when a child is of this GUI is being resized.</td>
</tr>
<tr>
<td>void</td>
<td>onGainFirstResponder (SimObjectId ID)</td>
<td>Called when this GUI gains focus.</td>
</tr>
<tr>
<td>void</td>
<td>onLoseFirstResponder (SimObjectId ID)</td>
<td>Called when this GUI loses focus.</td>
</tr>
<tr>
<td>void</td>
<td>onParentResized (SimObjectId ID)</td>
<td>Called when this GUI's parent is resized.</td>
</tr>
</tbody>
</table>
void onResize (SimObjectld ID)
Called when this GUI is resized.
Detailed Description

A control which adds several reactions to other GUIs via callbacks.

*GuiScriptNotifyCtrl* does not exist to render anything. When parented or made a child of other controls, you can toggle flags on or off to make use of its specialized callbacks. Normally these callbacks are used as utility functions by the GUI Editor, or other container classes. However, for very fancy GUI work where controls interact with each other constantly, this is a handy utility to make use of.

Example:

```java
// Common member fields left out for sake
new GuiScriptNotifyCtrl()
{
    onChildAdded = "0";
    onChildRemoved = "0";
    onChildResized = "0";
    onParentResized = "0";
};
```
**Member Function Documentation**

```cpp
void GuiScriptNotifyCtrl::onChildAdded (SimObjectId ID, SimObjectId childID)
```

Called when a child is added to this GUI.

**Parameters:**
- **ID** Unique object ID assigned when created (this in script).
- **childID** Unique object ID of child being added.

```cpp
void GuiScriptNotifyCtrl::onChildRemoved (SimObjectId ID, SimObjectId childID)
```

Called when a child is removed from this GUI.

**Parameters:**
- **ID** Unique object ID assigned when created (this in script).
- **childID** Unique object ID of child being removed.

```cpp
void GuiScriptNotifyCtrl::onChildResized (SimObjectId ID, SimObjectId childID)
```

Called when a child is of this GUI is being resized.

**Parameters:**
- **ID** Unique object ID assigned when created (this in script).
childID  Unique object ID of child being resized.

**void GuiScriptNotifyCtrl::onGainFirstResponder (SimObjectId ID )**

Called when this GUI gains focus.

**Parameters:**

*ID*  Unique object ID assigned when created (this in script).

**void GuiScriptNotifyCtrl::onLoseFirstResponder (SimObjectId ID )**

Called when this GUI loses focus.

**Parameters:**

*ID*  Unique object ID assigned when created (this in script).

**void GuiScriptNotifyCtrl::onParentResized (SimObjectId ID )**

Called when this GUI's parent is resized.

**Parameters:**

*ID*  Unique object ID assigned when created (this in script).

**void GuiScriptNotifyCtrl::onResize (SimObjectId ID )**

Called when this GUI is resized.

**Parameters:**

*ID*  Unique object ID assigned when created (this in script).
## Member Data Documentation

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bool GuiScriptNotifyCtrl::onChildAdded</code></td>
<td>Enables/disables <code>onChildAdded</code> callback.</td>
</tr>
<tr>
<td><code>bool GuiScriptNotifyCtrl::onChildRemoved</code></td>
<td>Enables/disables <code>onChildRemoved</code> callback.</td>
</tr>
<tr>
<td><code>bool GuiScriptNotifyCtrl::onChildResized</code></td>
<td>Enables/disables <code>onChildResized</code> callback.</td>
</tr>
<tr>
<td><code>bool GuiScriptNotifyCtrl::onGainFirstResponder</code></td>
<td>Enables/disables <code>onGainFirstResponder</code> callback.</td>
</tr>
<tr>
<td><code>bool GuiScriptNotifyCtrl::onLoseFirstResponder</code></td>
<td>Enables/disables <code>onLoseFirstResponder</code> callback.</td>
</tr>
<tr>
<td><code>bool GuiScriptNotifyCtrl::onParentResized</code></td>
<td>Enables/disables <code>onParentResized</code> callback.</td>
</tr>
<tr>
<td><code>bool GuiScriptNotifyCtrl::onResize</code></td>
<td>Enables/disables <code>onResize</code> callback.</td>
</tr>
</tbody>
</table>
GuiScrollCtrl Class Reference
[Container Controls]

A container that allows to view one or more possibly larger controls inside its area by providing horizontal and/or vertical scroll bars.

More...

Inheritance diagram for GuiScrollCtrl:

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>void</strong></td>
<td><code>computeSizes ()</code></td>
<td>Refresh sizing and positioning of child controls.</td>
</tr>
<tr>
<td><strong>Point2I</strong></td>
<td><code>getScrollPosition ()</code></td>
<td>Get the current coordinates of the scrolled content.</td>
</tr>
<tr>
<td><strong>int</strong></td>
<td><code>getScrollPositionX ()</code></td>
<td>Get the current X coordinate of the scrolled content.</td>
</tr>
<tr>
<td><strong>int</strong></td>
<td><code>getScrollPositionY ()</code></td>
<td>Get the current Y coordinate of the scrolled content.</td>
</tr>
<tr>
<td><strong>void</strong></td>
<td><code>scrollToBottom ()</code></td>
<td>Scroll all the way to the bottom of the vertical scrollbar and the left of the horizontal bar.</td>
</tr>
<tr>
<td><strong>void</strong></td>
<td><code>scrollToObject (GuiControl control)</code></td>
<td>Scroll the control so that the given child control is visible.</td>
</tr>
<tr>
<td><strong>void</strong></td>
<td><code>scrollToTop ()</code></td>
<td>Scroll all the way to the top of the vertical and left of the horizontal scrollbar.</td>
</tr>
<tr>
<td><strong>void</strong></td>
<td><code>setScrollPosition (int x, int y)</code></td>
<td>Set the position of the scrolled content.</td>
</tr>
</tbody>
</table>

### Callbacks

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>void</strong></td>
<td><code>onScroll ()</code></td>
<td>Called each time the child controls are scrolled by some amount.</td>
</tr>
</tbody>
</table>
Public Attributes

### Scrolling

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Point2I</code></td>
<td><code>childMargin</code></td>
<td>Padding region to put around child contents.</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><code>constantThumbHeight</code></td>
<td></td>
</tr>
<tr>
<td><code>GuiScrollBarBehavior</code></td>
<td><code>hScrollBar</code></td>
<td>When to display the horizontal scrollbar.</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><code>lockHorizScroll</code></td>
<td>Horizontal scrolling not allowed if set.</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><code>lockVertScroll</code></td>
<td>Vertical scrolling not allowed if set.</td>
</tr>
<tr>
<td><code>int</code></td>
<td><code>mouseWheelScrollSpeed</code></td>
<td>Pixels/Tick - if not positive then mousewheel scrolling occurs instantly (like other scrolling).</td>
</tr>
<tr>
<td><code>GuiScrollBarBehavior</code></td>
<td><code>vScrollBar</code></td>
<td>When to display the vertical scrollbar.</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><code>willFirstRespond</code></td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

A container that allows to view one or more possibly larger controls inside its area by providing horizontal and/or vertical scroll bars.
**Member Function Documentation**

```cpp
void GuiScrollCtrl::computeSizes( )
```

Refresh sizing and positioning of child controls.

```cpp
Point2I GuiScrollCtrl::getScrollPosition( )
```

Get the current coordinates of the scrolled content.

**Returns:**
The current position of the scrolled content.

```cpp
int GuiScrollCtrl::getScrollPositionX( )
```

Get the current X coordinate of the scrolled content.

**Returns:**
The current X coordinate of the scrolled content.

```cpp
int GuiScrollCtrl::getScrollPositionY( )
```

Get the current Y coordinate of the scrolled content.

**Returns:**
The current Y coordinate of the scrolled content.

```cpp
void GuiScrollCtrl::onScroll( )
```

Called each time the child controls are scrolled by some amount.
void GuiScrollCtrl::scrollToBottom( )

Scroll all the way to the bottom of the vertical scrollbar and the left of the horizontal bar.

void GuiScrollCtrl::scrollToObject(GuiControl control)

Scroll the control so that the given child control is visible.

**Parameters:**

control A child control.

void GuiScrollCtrl::scrollToTop( )

Scroll all the way to the top of the vertical and left of the horizontal scrollbar.

void GuiScrollCtrl::setScrollPosition(int x, int y)

Set the position of the scrolled content.

**Parameters:**

x Position on X axis.
y Position on y axis.
**Member Data Documentation**

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Point2I GuiScrollCtrl::childMargin</strong></td>
<td>Padding region to put around child contents.</td>
</tr>
<tr>
<td><strong>bool GuiScrollCtrl::constantThumbHeight</strong></td>
<td></td>
</tr>
<tr>
<td><strong>GuiScrollBarBehavior GuiScrollCtrl::hScrollBar</strong></td>
<td>When to display the horizontal scrollbar.</td>
</tr>
<tr>
<td><strong>bool GuiScrollCtrl::lockHorizScroll</strong></td>
<td>Horizontal scrolling not allowed if set.</td>
</tr>
<tr>
<td><strong>bool GuiScrollCtrl::lockVertScroll</strong></td>
<td>Vertical scrolling not allowed if set.</td>
</tr>
<tr>
<td><strong>int GuiScrollCtrl::mouseWheelScrollSpeed</strong></td>
<td>Pixels/Tick - if not positive then mousewheel scrolling occurs instantly (like other scrolling).</td>
</tr>
<tr>
<td><strong>GuiScrollBarBehavior GuiScrollCtrl::vScrollBar</strong></td>
<td>When to display the vertical scrollbar.</td>
</tr>
</tbody>
</table>
bool GuiScrollCtrl::willFirstRespond

Copyright © GarageGames, LLC. All Rights Reserved.
GuiSeparatorCtrl Class Reference
[General Controls]

A control that renders a horizontal or vertical separator with an optional text label (horizontal only). More...

Inheritance diagram for GuiSeparatorCtrl:

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><code>borderMargin</code></td>
<td></td>
</tr>
<tr>
<td>string</td>
<td><code>caption</code></td>
<td>Optional text label to display.</td>
</tr>
<tr>
<td>bool</td>
<td><code>invisible</code></td>
<td></td>
</tr>
<tr>
<td>int</td>
<td><code>leftMargin</code></td>
<td>Left margin of text label.</td>
</tr>
<tr>
<td></td>
<td><code>GuiSeparatorType</code></td>
<td>Orientation of separator.</td>
</tr>
</tbody>
</table>
Detailed Description

A control that renders a horizontal or vertical separator with an optional text label (horizontal only).

Example:

```java
new GuiSeparatorCtrl() {
    profile = "GuiDefaultProfile";
    position = "505 0";
    extent = "10 17";
    minExtent = "10 17";
    canSave = "1";
    visible = "1";
    horizSizing = "left";
};
```
## Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Member</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>GuiSeparatorCtrl::borderMargin</td>
<td></td>
</tr>
<tr>
<td>string</td>
<td>GuiSeparatorCtrl::caption</td>
<td>Optional text label to display.</td>
</tr>
<tr>
<td>bool</td>
<td>GuiSeparatorCtrl::invisible</td>
<td></td>
</tr>
<tr>
<td>int</td>
<td>GuiSeparatorCtrl::leftMargin</td>
<td>Left margin of text label.</td>
</tr>
<tr>
<td>GuiSeparatorType</td>
<td>GuiSeparatorCtrl::type</td>
<td>Orientation of separator.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
GuiShapeNameHud Class Reference
[Game Controls]

Displays name and damage of ShapeBase objects in its bounds. Must be a child of a GuiTSCtrl and a server connection must be present. More...

Inheritance diagram for GuiShapeNameHud:

List of all members.
## Public Attributes

### Misc

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>distanceFade</td>
<td>Visibility distance (how far the player must be from the ShapeBase object in focus) for this control to render.</td>
</tr>
<tr>
<td>bool</td>
<td>showFill</td>
<td>If true, we draw the background color of the control.</td>
</tr>
<tr>
<td>bool</td>
<td>showFrame</td>
<td>If true, we draw the frame of the control.</td>
</tr>
<tr>
<td>float</td>
<td>verticalOffset</td>
<td>Amount to vertically offset the control in relation to the ShapeBase object in focus.</td>
</tr>
</tbody>
</table>

### Colors

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorF</td>
<td>fillColor</td>
<td>Standard color for the background of the control.</td>
</tr>
<tr>
<td>ColorF</td>
<td>frameColor</td>
<td>Color for the control's frame.</td>
</tr>
<tr>
<td>ColorF</td>
<td>textColor</td>
<td>Color for the text on this control.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Displays name and damage of *ShapeBase* objects in its bounds. Must be a child of a *GuiTSCtrl* and a server connection must be present.

This control displays the name and damage value of all named *ShapeBase* objects on the client. The name and damage of objects within the control's display area are overlayed above the object.

This GUI control must be a child of a *TSControl*, and a server connection and control object must be present. This is a stand-alone control and relies only on the standard base *GuiControl*.

**Example:**

```java
new GuiShapeNameHud(){
    fillColor = "0.0 1.0 0.0 1.0"; // Fills
    frameColor = "1.0 1.0 1.0 1.0"; // Solid
    textColor = "1.0 1.0 1.0 1.0"; // Solid
    showFill = "true";
    showFrame = "true";
    verticalOffset = "0.15";
    distanceFade = "15.0";
};
```
### Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>GuiShapeNameHud::distanceFade</code></td>
</tr>
<tr>
<td></td>
<td>Visibility distance (how far the player must be from the <code>ShapeBase</code> object in focus) for this control to render.</td>
</tr>
<tr>
<td>ColorF</td>
<td><code>GuiShapeNameHud::fillColor</code></td>
</tr>
<tr>
<td></td>
<td>Standard color for the background of the control.</td>
</tr>
<tr>
<td>ColorF</td>
<td><code>GuiShapeNameHud::frameColor</code></td>
</tr>
<tr>
<td></td>
<td>Color for the control's frame.</td>
</tr>
<tr>
<td>bool</td>
<td><code>GuiShapeNameHud::showFill</code></td>
</tr>
<tr>
<td></td>
<td>If true, we draw the background color of the control.</td>
</tr>
<tr>
<td>bool</td>
<td><code>GuiShapeNameHud::showFrame</code></td>
</tr>
<tr>
<td></td>
<td>If true, we draw the frame of the control.</td>
</tr>
<tr>
<td>ColorF</td>
<td><code>GuiShapeNameHud::textColor</code></td>
</tr>
<tr>
<td></td>
<td>Color for the text on this control.</td>
</tr>
<tr>
<td>float</td>
<td><code>GuiShapeNameHud::verticalOffset</code></td>
</tr>
</tbody>
</table>
Amount to vertically offset the control in relation to the ShapeBase object in focus.

Copyright © GarageGames, LLC. All Rights Reserved.
GuiSliderCtrl Class Reference
[Value Controls]

A control that displays a value between its minimal and maximal bounds using a slider placed on a vertical or horizontal axis. More...

Inheritance diagram for GuiSliderCtrl:

```
SimObject
    ▼
    SimSet
        ▼
        SimGroup
            ▼
            GuiControl
                ▼
                GuiSliderCtrl
[legend]
```

List of all members.
Public Member Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>getValue ()</td>
<td>Get the current value of the slider based on the position of the thumb.</td>
</tr>
<tr>
<td>bool</td>
<td>isThumbBeingDragged ()</td>
<td>Returns true if the thumb is currently being dragged by the user. This method is mainly useful for scrubbing type sliders where the slider position is sync'd to a changing value. When the user is dragging the thumb, however, the sync'ing should pause and not get in the way of the user.</td>
</tr>
<tr>
<td>void</td>
<td>setValue (float pos, bool doCallback=false)</td>
<td>Set position of the thumb on the slider.</td>
</tr>
</tbody>
</table>

Callbacks

| void   | onMouseDragged ()                 | Called when the left mouse button is dragged across the slider.                                                                                   |
## Public Attributes

### Slider

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point2F</td>
<td>range</td>
<td>Min and max values corresponding to left and right slider position.</td>
</tr>
<tr>
<td>bool</td>
<td>snap</td>
<td>Whether to snap the slider to tick marks.</td>
</tr>
<tr>
<td>int</td>
<td>ticks</td>
<td>Spacing between tick marks in pixels. 0=off.</td>
</tr>
<tr>
<td>float</td>
<td>value</td>
<td>The value corresponding to the current slider position.</td>
</tr>
</tbody>
</table>
Detailed Description

A control that displays a value between its minimal and maximal bounds using a slider placed on a vertical or horizontal axis.

A slider displays a value and allows that value to be changed by dragging a thumb control along the axis of the slider. In this way, the value is changed between its allowed minimum and maximum.

To hook up script code to the value changes of a slider, use the command and altCommand properties. command is executed once the thumb is released by the user whereas altCommand is called any time the slider value changes. When changing the slider value from script, however, trigger of altCommand is suppressed by default.

The orientation of a slider is automatically determined from the ratio of its width to its height. If a slider is taller than it is wide, it will be rendered with a vertical orientation. If it is wider than it is tall, it will be rendered with a horizontal orientation.

The rendering of a slider depends on the bitmap in the slider's profile. This bitmap must be a bitmap array comprised of at least five bitmap rectangles. The rectangles are used such that:

- Rectangle #1: Left edge of slider
- Rectangle #2: Center piece of slider; this is stretched between the left and right edge
- Rectangle #3: Right edge of slider
- Rectangle #4: Thumb button in normal state
- Rectangle #5: Thumb button in highlighted (mouse-over) state

Example:

```plaintext
// Create a sound source and a slider that
%sound = sfxPlayOnce( "art/sound/testing"
new GuiSlider()
```
{  // Update the sound source volume when the slider is being dragged and released.
  command = %source @ ".setVolume( $ThisControl.value );"

  // Limit the range to 0..1 since that is the allowable range for sound volumes.
  range = "0 1";
};

See also:
GuiTextEditSliderCtrl
GuiTextEditSliderBitmapCtrl
Member Function Documentation

float GuiSliderCtrl::getValue ( )

Get the current value of the slider based on the position of the thumb.

**Returns:**
Slider position (from range.x to range.y).

bool GuiSliderCtrl::isThumbBeingDragged ( )

Returns true if the thumb is currently being dragged by the user. This method is mainly useful for scrubbing type sliders where the slider position is sync’d to a changing value. When the user is dragging the thumb, however, the sync’ing should pause and not get in the way of the user.

void GuiSliderCtrl::onMouseDragged ( )

Called when the left mouse button is dragged across the slider.

void GuiSliderCtrl::setValue (float pos,  
    bool doCallback = false  
)

Set position of the thumb on the slider.

**Parameters:**
- *pos*  New slider position (from range.x to range.y)
- *doCallback*  If true, the altCommand callback will be invoked
# Member Data Documentation

- **Point2F GuiSliderCtrl::range**
  
  Min and max values corresponding to left and right slider position.

- **bool GuiSliderCtrl::snap**
  
  Whether to snap the slider to tick marks.

- **int GuiSliderCtrl::ticks**
  
  Spacing between tick marks in pixels. 0=off.

- **float GuiSliderCtrl::value**
  
  The value corresponding to the current slider position.
**GuiSpeedometerHud Class Reference**

[Container Controls]

Displays the speed of the current **Vehicle** based control object.

**More...**

Inheritance diagram for GuiSpeedometerHud:

```
SimObject

SimSet

SimGroup

GuiControl

GuiBitmapCtrl

GuiSpeedometerHud
```

List of all members.
Public Attributes

Needle

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point2F</td>
<td>center</td>
<td>Center of the needle, offset from the GuiSpeedometerHud control top left corner.</td>
</tr>
<tr>
<td>ColorF</td>
<td>color</td>
<td>Color of the needle.</td>
</tr>
<tr>
<td>float</td>
<td>length</td>
<td>Length of the needle from center to end.</td>
</tr>
<tr>
<td>float</td>
<td>maxAngle</td>
<td>Angle (in radians) of the needle when the Vehicle speed is &gt;= maxSpeed. An angle of 0 points right, 90 points up etc).</td>
</tr>
<tr>
<td>float</td>
<td>maxSpeed</td>
<td>Maximum Vehicle speed (in Torque units per second) to represent on the speedo (Vehicle speeds greater than this are clamped to maxSpeed).</td>
</tr>
<tr>
<td>float</td>
<td>minAngle</td>
<td>Angle (in radians) of the needle when the Vehicle speed is 0. An angle of 0 points right, 90 points up etc).</td>
</tr>
<tr>
<td>float</td>
<td>tail</td>
<td>Length of the needle from center to tail.</td>
</tr>
<tr>
<td>float</td>
<td>width</td>
<td>Width of the needle.</td>
</tr>
</tbody>
</table>
Detailed Description

Displays the speed of the current Vehicle based control object.

This control only works if a server connection exists, and its control object is a Vehicle derived class. If either of these requirements is false, the control is not rendered.

The control renders the speedometer needle as a colored quad, rotated to indicate the Vehicle speed as determined by the minAngle, maxAngle, and maxSpeed properties. This control is normally placed on top of a GuiBitmapCtrl representing the speedometer dial.

Example:

```
new GuiSpeedometerHud()
{
    maxSpeed = "100";
    minAngle = "215";
    maxAngle = "0";
    color = "1 0.3 0.3 1";
    center = "130 123";
    length = "100";
    width = "2";
    tail = "0";
    //Properties not specific to this control
};
```
## Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Point2F</code></td>
<td><code>GuiSpeedometerHud::center</code></td>
</tr>
<tr>
<td></td>
<td>Center of the needle, offset from the <code>GuiSpeedometerHud</code> control top left corner.</td>
</tr>
<tr>
<td><code>ColorF</code></td>
<td><code>GuiSpeedometerHud::color</code></td>
</tr>
<tr>
<td></td>
<td>Color of the needle.</td>
</tr>
<tr>
<td><code>float</code></td>
<td><code>GuiSpeedometerHud::length</code></td>
</tr>
<tr>
<td></td>
<td>Length of the needle from center to end.</td>
</tr>
<tr>
<td><code>float</code></td>
<td><code>GuiSpeedometerHud::maxAngle</code></td>
</tr>
<tr>
<td></td>
<td>Angle (in radians) of the needle when the Vehicle speed is $\geq$ maxSpeed. An angle of 0 points right, 90 points up etc).</td>
</tr>
<tr>
<td><code>float</code></td>
<td><code>GuiSpeedometerHud::maxSpeed</code></td>
</tr>
<tr>
<td></td>
<td>Maximum Vehicle speed (in Torque units per second) to represent on the speedo (Vehicle speeds greater than this are clamped to maxSpeed).</td>
</tr>
<tr>
<td><code>float</code></td>
<td><code>GuiSpeedometerHud::minAngle</code></td>
</tr>
<tr>
<td></td>
<td>Angle (in radians) of the needle when the Vehicle speed is 0. An angle of 0 points right, 90 points up etc).</td>
</tr>
</tbody>
</table>
float GuiSpeedometerHud::tail

Length of the needle from center to tail.

float GuiSpeedometerHud::width

Width of the needle.
GuiSplitContainer Class Reference
[Container Controls]

A container that splits its area between two child controls. More...

Inheritance diagram for GuiSplitContainer:

```
  SimObject
   ^
  SimSet
   |
  SimGroup
   |
GuiControl
   |
GuiContainer
   |
GuiSplitContainer
```

List of all members.
## Public Attributes

### Splitter

*Options to configure split panels contained by this control*

<table>
<thead>
<tr>
<th>Class</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GuiSplitFixedPanel</td>
<td>fixedPanel</td>
<td>Which (if any) side of the splitter to keep at a fixed size.</td>
</tr>
<tr>
<td>int</td>
<td>fixedSize</td>
<td>Width of the fixed panel specified by fixedPanel (if any).</td>
</tr>
<tr>
<td>GuiSplitOrientation</td>
<td>orientation</td>
<td>Whether to split between top and bottom (horizontal) or between left and right (vertical).</td>
</tr>
<tr>
<td>Point2I</td>
<td>splitPoint</td>
<td>Point on control through which the splitter goes.</td>
</tr>
<tr>
<td>int</td>
<td>splitterSize</td>
<td>Width of the splitter bar between the two sides. Default is 2.</td>
</tr>
</tbody>
</table>
**Detailed Description**

A container that splits its area between two child controls.

A **GuiSplitContainer** can be used to dynamically subdivide an area between two child controls. A splitter bar is placed between the two controls and allows to dynamically adjust the sizing ratio between the two sides. Splitting can be either horizontal (subdividing top and bottom) or vertical (subdividing left and right) depending on orientation.

By using **fixedPanel**, one of the panels can be chosen to remain at a fixed size (**fixedSize**).

**Example:**

```java
// Create a vertical splitter with a fixed size
s = new GuiSplitContainer()
{
    orientation = "Vertical";
    fixedPanel = "FirstPanel";
    fixedSize = 100;

    new GuiScrollCtrl()
    {
        new GuiMLTextCtrl()
        {
            text = %longText;
        }
    }

    new GuiScrollCtrl()
    {
        new GuiMLTextCtrl()
        {
            text = %moreLongText;
        }
    }
}
```
Note:
The children placed inside GuiSplitContainers must be GuiContainers.
**Member Data Documentation**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GuiSplitFixedPanel GuiSplitContainer::fixedPanel</td>
<td>Which (if any) side of the splitter to keep at a fixed size.</td>
</tr>
<tr>
<td>int GuiSplitContainer::fixedSize</td>
<td>Width of the fixed panel specified by fixedPanel (if any).</td>
</tr>
<tr>
<td>GuiSplitOrientation GuiSplitContainer::orientation</td>
<td>Whether to split between top and bottom (horizontal) or between left and right (vertical).</td>
</tr>
<tr>
<td>Point2I GuiSplitContainer::splitPoint</td>
<td>Point on control through which the splitter goes.</td>
</tr>
<tr>
<td></td>
<td>Changed relatively if size of control changes.</td>
</tr>
<tr>
<td>int GuiSplitContainer::splitterSize</td>
<td>Width of the splitter bar between the two sides. Default is 2.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
GuiStackControl Class Reference
[Container Controls]

A container that stacks its children horizontally or vertically. More...

Inheritance diagram for GuiStackControl:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void freeze (bool freeze)</code></td>
<td>Prevents control from restacking - useful when adding or removing child controls.</td>
</tr>
<tr>
<td><code>bool isFrozen ()</code></td>
<td>Return whether or not this control is frozen.</td>
</tr>
<tr>
<td><code>void updateStack ()</code></td>
<td>Restack the child controls.</td>
</tr>
</tbody>
</table>
## Public Attributes

### Stacking

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>changeChildPosition</code></td>
<td>Determines whether to reposition child controls.</td>
</tr>
<tr>
<td>bool</td>
<td><code>changeChildSizeToFit</code></td>
<td>Determines whether to resize child controls.</td>
</tr>
<tr>
<td>bool</td>
<td><code>dynamicNonStackExtent</code></td>
<td>Determines whether to resize the stack control along the non-stack axis (change height for horizontal stacking, change width for vertical stacking). No effect if <code>dynamicSize</code> is false.</td>
</tr>
<tr>
<td>bool</td>
<td><code>dynamicPos</code></td>
<td>Determines whether to reposition the stack along the stack axis when it is auto-resized. No effect if <code>dynamicSize</code> is false.</td>
</tr>
<tr>
<td>bool</td>
<td><code>dynamicSize</code></td>
<td>Determines whether to resize the stack control along the stack axis (change width for horizontal stacking, change height for vertical stacking).</td>
</tr>
<tr>
<td>int</td>
<td><code>padding</code></td>
<td>Distance (in pixels) between stacked child controls.</td>
</tr>
<tr>
<td>GuiHorizontalStackingType</td>
<td><code>horizStacking</code></td>
<td>Controls the type of horizontal stacking to use (<em>Left to Right</em> or <em>Right to Left</em>).</td>
</tr>
<tr>
<td><strong>GuiStackingType</strong></td>
<td><strong>stackingType</strong></td>
<td></td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Determines the method used to</td>
<td>position the child controls.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>GuiVerticalStackingType</strong></td>
<td><strong>vertStacking</strong></td>
<td></td>
</tr>
<tr>
<td>Controls the type of vertical</td>
<td>Controls the type of vertical stacking</td>
<td></td>
</tr>
<tr>
<td>stacking to use <em>(Top to Bottom</em></td>
<td>to use <em>(Top to Bottom or Bottom to Top)</em></td>
<td></td>
</tr>
<tr>
<td>or Bottom to Top)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

A container that stacks its children horizontally or vertically.

This container maintains a horizontal or vertical stack of GUI controls. If one is added, deleted, or resized, then the stack is resized to fit. The order of the stack is determined by the internal order of the children (ie. the order of addition).

Example:

```java
new GuiStackControl()
{
    stackingType = "Dynamic";
    horizStacking = "Left to Right";
    vertStacking = "Top to Bottom";
    padding = "4";
    dynamicSize = "1";
    dynamicNonStackExtent = "0";
    dynamicPos = "0";
    changeChildSizeToFit = "1";
    changeChildPosition = "1";
    //Properties not specific to this control
}
```
### Member Function Documentation

**void GuiStackControl::freeze (bool freeze )**

Prevents control from restacking - useful when adding or removing child controls.

**Parameters:**

freeze True to freeze the control, false to unfreeze it

**Example:**

```cpp
%stackCtrl.freeze(true);
// add controls to stack
%stackCtrl.freeze(false);
```

**bool GuiStackControl::isFrozen ( )**

Return whether or not this control is frozen.

**void GuiStackControl::updateStack ( )**

Restack the child controls.
## Member Data Documentation

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| **bool GuiStackControl::changeChildPosition** | Determines whether to reposition child controls.  
If true, horizontally stacked children are aligned along the top edge of the stack control. Vertically stacked children are aligned along the left edge of the stack control. If false, horizontally stacked children retain their Y position, and vertically stacked children retain their X position. |
| **bool GuiStackControl::changeChildSizeToFit** | Determines whether to resize child controls.  
If true, horizontally stacked children keep their width, but have their height set to the stack control height. Vertically stacked children keep their height, but have their width set to the stack control width. If false, child controls are not resized. |
| **bool GuiStackControl::dynamicNonStackExtent** | Determines whether to resize the stack control along the non-stack axis (change height for horizontal stacking, change width for vertical stacking). No effect if dynamicSize is false.  
If true, the stack will be resized to the maximum of the child control widths/heights. If false, the stack will not be resized. |
| **bool GuiStackControl::dynamicPos** | Determines whether to reposition the stack along the stack axis |
when it is auto-resized. No effect if dynamicSize is false.

If true, the stack will grow left for horizontal stacking, and grow up for vertical stacking. If false, the stack will grow right for horizontal stacking, and grow down for vertical stacking.

**bool GuiStackControl::dynamicSize**

Determines whether to resize the stack control along the stack axis (change width for horizontal stacking, change height for vertical stacking).

If true, the stack width/height will be resized to the sum of the child control widths/heights. If false, the stack will not be resized.

**GuiHorizontalStackingType GuiStackControl::horizStacking**

Controls the type of horizontal stacking to use (*Left to Right* or *Right to Left*).

**int GuiStackControl::padding**

Distance (in pixels) between stacked child controls.

**GuiStackingType GuiStackControl::stackingType**

Determines the method used to position the child controls.

**GuiVerticalStackingType GuiStackControl::vertStacking**

Controls the type of vertical stacking to use (*Top to Bottom* or *Bottom to Top*).
Copyright © GarageGames, LLC. All Rights Reserved.
GuiSwatchButtonCtrl Class Reference

[Button Controls]

A button that is used to represent color; often used in correlation with a color picker. More...

Inheritance diagram for GuiSwatchButtonCtrl:

```
SimObject
  
SimSet
  
SimGroup
  
GuiControl
  
GuiButtonBaseCtrl
  
GuiSwatchButtonCtrl
```

[legend]

List of all members.
Public Member Functions

```c
void setColor (string newColor)
  Set the color of the swatch control.
```
### Public Attributes

<table>
<thead>
<tr>
<th>ColorF</th>
<th>color</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The foreground color of <code>GuiSwatchButtonCtrl</code>.</td>
</tr>
</tbody>
</table>
**Detailed Description**

A button that is used to represent color; often used in correlation with a color picker.

A swatch button is a push button that uses its color field to designate the color drawn over an image, on top of a button.

The color itself is a float value stored inside the `GuiSwatchButtonCtrl::color` field. The texture path that represents the image underlying the color is stored inside the `GuiSwatchButtonCtrl::bitmap` field. The default value assigned to `GuiSwatchButtonCtrl::color` is "1 1 1 1" (White). The default/fallback image assigned to `GuiSwatchButtonCtrl::bitmap` is "core/art/gui/images/transp_grid".

**Example:**

```cpp
// Create a GuiSwatchButtonCtrl that calls randomFunction when clicked
%swatchButton = new GuiSwatchButtonCtrl()
{
    profile = "GuiInspectorSwatchButtonProfile"
    command = "randomFunction( $ThisControl.color );"
};
```
Set the color of the swatch control.

**Parameters:**

- `newColor` The new color string given to the swatch control in float format "r g b a".

**Note:**

It's also important to note that when setColor is called causes the control's altCommand field to be executed.
Member Data Documentation

ColorF GuiSwatchButtonCtrl::color

The foreground color of GuiSwatchButtonCtrl.

Copyright © GarageGames, LLC. All Rights Reserved.
GuiTabBookCtrl Class Reference
[Container Controls]

A container. More...

Inheritance diagram for GuiTabBookCtrl:

List of all members.
Public Member Functions

void **addPage** (string title="")
Add a new tab page to the control.

int **getSelectedPage** ()
Get the index of the currently selected tab page.

void **selectPage** (int index)
Set the selected tab page.

Callbacks

void **onTabRightClick** (String text, int index)
Called when the user right-clicks on a tab page header.

void **onTabSelected** (String text, int index)
Called when a new tab page is selected.
# Public Attributes

## TabBook

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>allowReorder</td>
<td>Whether reordering tabs with the mouse is allowed.</td>
</tr>
<tr>
<td>int</td>
<td>defaultPage</td>
<td>Index of page to select on first <code>onWake()</code> call (-1 to disable).</td>
</tr>
<tr>
<td>int</td>
<td>frontTabPadding</td>
<td>X offset of first tab page header.</td>
</tr>
<tr>
<td>int</td>
<td>minTabWidth</td>
<td>Minimum width allocated to a tab page header.</td>
</tr>
<tr>
<td>int</td>
<td>selectedPage</td>
<td>Index of currently selected page.</td>
</tr>
<tr>
<td>int</td>
<td>tabHeight</td>
<td>Height of tab page headers.</td>
</tr>
<tr>
<td>int</td>
<td>tabMargin</td>
<td>Spacing to put between individual tab page headers.</td>
</tr>
<tr>
<td>GuiTabPosition</td>
<td>tabPosition</td>
<td>Where to place the tab page headers.</td>
</tr>
</tbody>
</table>
Detailed Description

A container.

Example:

```c
// Create
```

Note:

Only GuiTabPageCtrls must be added to GuiTabBookCtrls. If an object of a different class is added to the control, it will be reassigned to either the active page or the tab book’s parent.

See also:

`GuiTabPageCtrl`
Member Function Documentation

void GuiTabBookCtrl::addPage (string title = "")

Add a new tab page to the control.

Parameters:

   title Title text for the tab page header.

int GuiTabBookCtrl::getSelectedPage ( )

Get the index of the currently selected tab page.

Returns:

   Index of the selected tab page or -1 if no tab page is selected.

void GuiTabBookCtrl::onTabRightClick (String text,
                                          int index )

Called when the user right-clicks on a tab page header.

Parameters:

   text Text of the page header for the tab that is being selected.

   index Index of the tab page being selected.

void GuiTabBookCtrl::onTabSelected (String text,
                                         int index )

Called when a new tab page is selected.
Parameters:

- **text** Text of the page header for the tab that is being selected.
- **index** Index of the tab page being selected.

```cpp
void GuiTabBookCtrl::selectPage(int index)
```

Set the selected tab page.

Parameters:

- **index** Index of the tab page.
## Member Data Documentation

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool GuiTabBookCtrl::allowReorder</td>
<td>Whether reordering tabs with the mouse is allowed.</td>
</tr>
<tr>
<td>int GuiTabBookCtrl::defaultPage</td>
<td>Index of page to select on first <code>onWake()</code> call (-1 to disable).</td>
</tr>
<tr>
<td>int GuiTabBookCtrl::frontTabPadding</td>
<td>X offset of first tab page header.</td>
</tr>
<tr>
<td>int GuiTabBookCtrl::minTabWidth</td>
<td>Minimum width allocated to a tab page header.</td>
</tr>
<tr>
<td>int GuiTabBookCtrl::selectedPage</td>
<td>Index of currently selected page.</td>
</tr>
<tr>
<td>int GuiTabBookCtrl::tabHeight</td>
<td>Height of tab page headers.</td>
</tr>
<tr>
<td>int GuiTabBookCtrl::tabMargin</td>
<td>Spacing to put between individual tab page headers.</td>
</tr>
</tbody>
</table>
GuiTabPosition GuiTabBookCtrl::tabPosition

Where to place the tab page headers.
GuiTabPageCtrl Class Reference
[Container Controls]

A single page in a GuiTabBookCtrl. More...

Inheritance diagram for GuiTabPageCtrl:

```
SimObject
  |
  |
SimSet
  |
  |
SimGroup
  |
  |
GuiControl
  |
  |
GuiContainer
  |
  |
GuiTextCtrl
  |
GuiTabPageCtrl
```

List of all members.
Public Member Functions

void select ()

Select this page in its tab book.
Public Attributes

bool fitBook

Determines whether to resize this page when it is added to the tab book. If true, the page will be resized according to the tab book extents and tabPosition property.
Detailed Description

A single page in a GuiTabBookCtrl.

Example:

```java
new GuiTabPageCtrl()
{
    fitBook = "1";
    //Properties not specific to this control
};
```
void GuiTabPageCtrl::select(

Select this page in its tab book.


Member Data Documentation

bool GuiTabPageCtrl::fitBook

Determines whether to resize this page when it is added to the tab book. If true, the page will be resized according to the tab book extents and tabPosition property.
GuiTextCtrl Class Reference
[Core Controls]

GUI control object this displays a single line of text, without TorqueML. More...

Inheritance diagram for GuiTextCtrl:

[legend]

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void setText (string text)</code></td>
<td>Sets the text in the control.</td>
</tr>
<tr>
<td><code>void setTextID (string textID)</code></td>
<td>Maps the text ctrl to a variable used in localization, rather than raw text.</td>
</tr>
</tbody>
</table>
### Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><code>maxLength</code></td>
<td>Defines the maximum length of the text. The default is 1024.</td>
</tr>
<tr>
<td>caseString</td>
<td><code>text</code></td>
<td>The text to show on the control.</td>
</tr>
<tr>
<td>string</td>
<td><code>textID</code></td>
<td>Maps the text of this control to a variable used in localization, rather than raw text.</td>
</tr>
</tbody>
</table>
Detailed Description

GUI control object this displays a single line of text, without TorqueML.

Example:

```java
new GuiTextCtrl()
{
    text = "Hello World";
    textID = "STR_HELLO";
    maxlength = "1024";
    //Properties not specific to this control have been omitted from this example.
}
```

See also:

- GuiControl
- Localization
void GuiTextCtrl::setText(string text)

Sets the text in the control.

**Parameters:**

*text* Text to display in the control.

**Example:**

```cpp
// Set the text to show in the control
%text = "Gideon - Destroyer of World";

// Inform the GuiTextCtrl control to change
%thisGuiTextCtrl.setText(%text);
```

**See also:**

GuiControl

Reimplemented in GuiTextEditCtrl, and GuiPopupMenuCtrlEx.

void GuiTextCtrl::setTextID(string textID)

Maps the text ctrl to a variable used in localization, rather than raw text.

**Parameters:**

*textID* Name of variable text should be mapped to

**Example:**

```cpp
// Inform the GuiTextCtrl control of the text ID
%thisGuiTextCtrl.setTextID("STR_QUIT");
```
See also:

GuiControl
Localization
### Member Data Documentation

**int GuiTextCtrl::maxLength**

Defines the maximum length of the text. The default is 1024.

**caseString GuiTextCtrl::text**

The text to show on the control.

**string GuiTextCtrl::textID**

Maps the text of this control to a variable used in localization, rather than raw text.
GuiTextEditCtrl Class Reference
[General Controls]

A component that places a text entry box on the screen. More...

Inheritance diagram for GuiTextEditCtrl:

```
SimObject

SimSet

SimGroup

GuiControl

GuiContainer

GuiTextEditCtrl

GuiTextEditCtrl

GuiConsoleEditCtrl  GuiTextEditSliderBitmapCtrl  GuiTextEditSliderCtrl
```

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void clearSelectedText()</td>
<td>Unselects all selected text in the control.</td>
</tr>
<tr>
<td>void forceValidateText()</td>
<td>Force a validation to occur.</td>
</tr>
<tr>
<td>int getCursorPos()</td>
<td>Returns the current position of the text cursor in the control.</td>
</tr>
<tr>
<td>string getText()</td>
<td>Acquires the current text displayed in this control.</td>
</tr>
<tr>
<td>bool isAllTextSelected()</td>
<td>Checks to see if all text in the control has been selected.</td>
</tr>
<tr>
<td>void selectAllText()</td>
<td>Selects all text within the control.</td>
</tr>
<tr>
<td>void setCursorPos(int position)</td>
<td>Sets the text cursor at the defined position within the control.</td>
</tr>
<tr>
<td>void setText (string text)</td>
<td>Sets the text in the control.</td>
</tr>
</tbody>
</table>

## Callbacks

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void onReturn ()</td>
<td>Called when the 'Return' or 'Enter' key is pressed.</td>
</tr>
<tr>
<td>void onTabComplete (string val)</td>
<td>Called if tabComplete is true, and the 'tab' key is pressed.</td>
</tr>
<tr>
<td>void onValidate ()</td>
<td>Called whenever the control is validated.</td>
</tr>
</tbody>
</table>
## Public Attributes

### Text Input

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SFXTrack</td>
<td>deniedSound</td>
<td>If the attempted text cannot be entered, this sound effect will be played.</td>
</tr>
<tr>
<td>string</td>
<td>escapeCommand</td>
<td>Script command to be called when the Escape key is pressed.</td>
</tr>
<tr>
<td>int</td>
<td>historySize</td>
<td>How large of a history buffer to maintain.</td>
</tr>
<tr>
<td>bool</td>
<td>password</td>
<td>If true, all characters entered will be stored in the control, however will display as the character stored in passwordMask.</td>
</tr>
<tr>
<td>string</td>
<td>passwordMask</td>
<td>If 'password' is true, this is the character that will be used to mask the characters in the control.</td>
</tr>
<tr>
<td>bool</td>
<td>sinkAllKeyEvents</td>
<td>If true, every key event will act as if the Enter key was pressed.</td>
</tr>
<tr>
<td>bool</td>
<td>tabComplete</td>
<td>If true, when the 'tab' key is pressed, it will act as if the Enter key was pressed on the control.</td>
</tr>
<tr>
<td>string</td>
<td>validate</td>
<td>Script command to be called when the first validator is lost.</td>
</tr>
</tbody>
</table>
**Detailed Description**

A component that places a text entry box on the screen.

Fonts and sizes are changed using profiles. The text value can be set or entered by a user.

**Example:**

```java
new GuiTextEditCtrl(MessageHud_Edit) {
    text = "Hello World";
    validate = "validateCommand();";
    escapeCommand = "escapeCommand();";
    historySize = "5";
    tabComplete = "true";
    deniedSound = "DeniedSoundProfile";
    sinkAllKeyEvents = "true";
    password = "true";
    passwordMask = "*";
    //Properties not specific to this
};
```

**See also:**

GuiTextCtrl
GuiControl
Member Function Documentation

void GuiTextEditCtrl::clearSelectedText( )

Unselects all selected text in the control.

Example:

    // Inform the control to unselect all of:
    %thisGuiTextEditCtrl.clearSelectedText();

See also:
    GuiControl

void GuiTextEditCtrl::forceValidateText( )

Force a validation to occur.

Example:

    // Inform the control to force a validation
    %thisGuiTextEditCtrl.forceValidateText();

See also:
    GuiControl

int GuiTextEditCtrl::getCursorPos( )

Returns the current position of the text cursor in the control.

Example:

    // Acquire the cursor position in the control
    %position = %thisGuiTextEditCtrl.getCursorPosition();
Returns:
Text cursor position within the control.

See also:
GuiControl

string GuiTextEditCtrl::getText()  

Acquires the current text displayed in this control.

Example:

// Acquire the value of the text control.
%text = %thisGuiTextEditCtrl.getText();

Returns:
The current text within the control.

See also:
GuiControl

bool GuiTextEditCtrl::isAllTextSelected()  

Checks to see if all text in the control has been selected.

Example:

// Check to see if all text has been selected.
%allSelected = %thisGuiTextEditCtrl.isAllTextSelected();

Returns:
True if all text in the control is selected, otherwise false.

See also:
GuiControl
void GuiTextEditCtrl::onReturn( )

Called when the 'Return' or 'Enter' key is pressed.

Example:

```cpp
// Return or Enter key was pressed, causing the callback to occur.
GuiTextEditCtrl::onReturn(%this)
{
    // Code to run when the onReturn callback occurs
}
```

See also:
GuiTextCtrl
GuiControl

---

void GuiTextEditCtrl::onTabComplete(string val)

Called if tabComplete is true, and the 'tab' key is pressed.

Parameters:

- `val` Input to mimick the '1' sent by the actual tab key button press.

Example:

```cpp
// Tab key has been pressed, causing the callback to occur.
GuiTextEditCtrl::onTabComplete(%this,%val)
{
    // Code to run when the onTabComplete callback occurs
}
```

See also:
void GuiTextEditCtrl::onValidate()

Called whenever the control is validated.

Example:

```
// The control gets validated, causing the
GuiTextEditCtrl::onValidated(%this)
{
    // Code to run when the control is \n
}
```

See also:
GuiTextCtrl
GuiControl

void GuiTextEditCtrl::selectAllText()

Selects all text within the control.

Example:

```
// Inform the control to select all of its
%thisGuiTextEditCtrl.selectAllText();
```

See also:
GuiControl

void GuiTextEditCtrl::setCursorPos(int position)

Sets the text cursor at the defined position within the control.
Parameters:

position  Text position to set the text cursor.

Example:

```cpp
// Define the cursor position
%position = "12";

// Inform the GuiTextEditCtrl control to place the text cursor at the defined position
%thisGuiTextEditCtrl.setCursorPos(%position);
```

See also:

GuiControl

---

```cpp
void GuiTextEditCtrl::setText(string text )
```

Sets the text in the control.

Parameters:

text  Text to place in the control.

Example:

```cpp
// Define the text to display
%text = "Text!"

// Inform the GuiTextEditCtrl to display the defined text
%thisGuiTextEditCtrl.setText(%text);
```

See also:

GuiControl

Reimplemented from GuiTextCtrl.
### Member Data Documentation

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SFXTrack</td>
<td>GuiTextEditCtrl::deniedSound</td>
<td>If the attempted text cannot be entered, this sound effect will be played.</td>
</tr>
<tr>
<td>string</td>
<td>GuiTextEditCtrl::escapeCommand</td>
<td>Script command to be called when the Escape key is pressed.</td>
</tr>
<tr>
<td>int</td>
<td>GuiTextEditCtrl::historySize</td>
<td>How large of a history buffer to maintain.</td>
</tr>
<tr>
<td>bool</td>
<td>GuiTextEditCtrl::password</td>
<td>If true, all characters entered will be stored in the control, however will display as the character stored in passwordMask.</td>
</tr>
<tr>
<td>string</td>
<td>GuiTextEditCtrl::passwordMask</td>
<td>If 'password' is true, this is the character that will be used to mask the characters in the control.</td>
</tr>
<tr>
<td>bool</td>
<td>GuiTextEditCtrl::sinkAllKeyEvents</td>
<td>If true, every key event will act as if the Enter key was pressed.</td>
</tr>
</tbody>
</table>
bool GuiTextEditCtrl::tabComplete

If true, when the 'tab' key is pressed, it will act as if the Enter key was pressed on the control.

string GuiTextEditCtrl::validate

Script command to be called when the first validater is lost.
GuiTextEditSliderBitmapCtrl Class Reference
[Core Controls]

GUI Control which displays a numerical value which can be increased or decreased using a pair of bitmap up/down buttons.

More...

Inheritance diagram for GuiTextEditSliderBitmapCtrl:

```
[legend]
```

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filename</td>
<td>bitmap</td>
<td>Unused.</td>
</tr>
<tr>
<td>bool</td>
<td>focusOnMouseWheel</td>
<td>If true, the control will accept giving focus to the user when the mouse wheel is used.</td>
</tr>
<tr>
<td>string</td>
<td>format</td>
<td>Character format type to place in the control.</td>
</tr>
<tr>
<td>float</td>
<td>increment</td>
<td>How far to increment the slider on each step.</td>
</tr>
<tr>
<td>Point2F</td>
<td>range</td>
<td>Maximum vertical and horizontal range to allow in the control.</td>
</tr>
</tbody>
</table>
**Detailed Description**

GUI Control which displays a numerical value which can be increased or decreased using a pair of bitmap up/down buttons.

This control uses the bitmap specified in it’s profile (GuiControlProfile::bitmapName). It takes this image and breaks up aspects of it to render the up and down arrows. It is also important to set GuiControlProfile::hasBitmapArray to true on the profile as well.

The bitmap referenced should be broken up into a 1 x 4 grid (using the top left color pixel as a border color between each of the images) in which it will map to the following places:

1. Up arrow active
2. Up arrow inactive
3. Down arrow active
4. Down arrow inactive

**Example:**

```cpp
singleton GuiControlProfile (SliderBitmapGUIProfile)
{
  bitmap = "core/art/gui/images/sliderArray"
  hasBitmapArray = true;
  opaque = false;
};

new GuiTextEditSliderBitmapCtrl()
{
  profile = "SliderBitmapGUIProfile";
  format = "%3.2f";
  range = "-1e+03 1e+03";
};
```
increment = "0.1";
focusOnMouseWheel = "0";
bitmap = "";

//Properties not specific to this control

See also:

GuiTextEditSliderCtrl
GuiTextEditCtrl
Member Data Documentation

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filename</td>
<td>GuiTextEditSliderBitmapCtrl::bitmap</td>
<td>Unused.</td>
</tr>
<tr>
<td>bool</td>
<td>GuiTextEditSliderBitmapCtrl::focusOnMouseWheel</td>
<td>If true, the control will accept giving focus to the user when the mouse wheel is used.</td>
</tr>
<tr>
<td>string</td>
<td>GuiTextEditSliderBitmapCtrl::format</td>
<td>Character format type to place in the control.</td>
</tr>
<tr>
<td>float</td>
<td>GuiTextEditSliderBitmapCtrl::increment</td>
<td>How far to increment the slider on each step.</td>
</tr>
<tr>
<td>Point2F</td>
<td>GuiTextEditSliderBitmapCtrl::range</td>
<td>Maximum vertical and horizontal range to allow in the control.</td>
</tr>
</tbody>
</table>
GuiTextEditSliderCtrl Class Reference
[Core Controls]

GUI Control which displays a numerical value which can be increased or decreased using a pair of arrows. More...

Inheritance diagram for GuiTextEditSliderCtrl:

```
SimObject
    ▼
    SimSet
    ▼
    SimGroup
    ▼
    GuiControl
    ▼
    GuiContainer
    ▼
    GuiTextCtrl
    ▼
    GuiTextEditCtrl
    ▼
    GuiTextEditSliderCtrl
```

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>focusOnMouseWheel</td>
<td>If true, the control will accept giving focus to the user when the mouse wheel is used.</td>
</tr>
<tr>
<td>string</td>
<td>format</td>
<td>Character format type to place in the control.</td>
</tr>
<tr>
<td>float</td>
<td>increment</td>
<td>How far to increment the slider on each step.</td>
</tr>
<tr>
<td>Point2F</td>
<td>range</td>
<td>Maximum vertical and horizontal range to allow in the control.</td>
</tr>
</tbody>
</table>
Detailed Description

GUI Control which displays a numerical value which can be increased or decreased using a pair of arrows.

Example:

```java
new GuiTextEditSliderCtrl()
{
    format = "%3.2f";
    range = "-1e+03 1e+03";
    increment = "0.1";
    focusOnMouseWheel = "0";
    //Properties not specific to this control
}
```

See also:

GuiTextEditCtrl
Member Data Documentation

**bool GuiTextEditSliderCtrl::focusOnMouseWheel**

If true, the control will accept giving focus to the user when the mouse wheel is used.

**string GuiTextEditSliderCtrl::format**

Character format type to place in the control.

**float GuiTextEditSliderCtrl::increment**

How far to increment the slider on each step.

**Point2F GuiTextEditSliderCtrl::range**

Maximum vertical and horizontal range to allow in the control.
GUI control that displays a list of text. Text items in the list can be individually selected. More...

Inheritance diagram for GuiTextListCtrl:

```
          GuiArrayCtrl
            ▼
GuiTextListCtrl
```

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>int addRow (int id=0, string text=&quot;&quot;, int index=-1)</strong></td>
<td>Adds a new row at end of the list with the defined id and text. If index is used, then the new row is inserted at the row location of 'index'.</td>
</tr>
<tr>
<td><strong>void clear ()</strong></td>
<td>Clear the list.</td>
</tr>
<tr>
<td><strong>void clearSelection ()</strong></td>
<td>Set the selection to nothing.</td>
</tr>
<tr>
<td><strong>int findTextIndex (string needle)</strong></td>
<td>Find needle in the list, and return the row number it was found in.</td>
</tr>
<tr>
<td><strong>int getRowId (int index)</strong></td>
<td>Get the row ID for an index.</td>
</tr>
<tr>
<td><strong>int getRowNumById (int id)</strong></td>
<td>Get the row number for a specified id.</td>
</tr>
<tr>
<td><strong>string getRowText (int index)</strong></td>
<td>Get the text of the row with the specified index.</td>
</tr>
<tr>
<td><strong>string getRowTextById (int id)</strong></td>
<td>Get the text of a row with the specified id.</td>
</tr>
<tr>
<td><strong>int getSelectedId ()</strong></td>
<td>Get the ID of the currently selected item.</td>
</tr>
<tr>
<td><strong>int getSelectedRow ()</strong></td>
<td>Returns the selected row index (not the row ID).</td>
</tr>
<tr>
<td><strong>bool isRowActive (int rowNum)</strong></td>
<td>Check if the specified row is currently active or not.</td>
</tr>
<tr>
<td><strong>void removeRow (int index)</strong></td>
<td>Remove a row from the table, based on its index.</td>
</tr>
<tr>
<td><strong>void removeRowById (int id)</strong></td>
<td>Remove row with the specified id.</td>
</tr>
</tbody>
</table>
### Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int  rowCount ()</td>
<td>Get the number of rows.</td>
</tr>
<tr>
<td>void scrollVisible (int rowNum)</td>
<td>Scroll so the specified row is visible.</td>
</tr>
<tr>
<td>void setRowActive (int rowNum, bool active)</td>
<td>Mark a specified row as active/not.</td>
</tr>
<tr>
<td>void setRowById (int id, string text)</td>
<td>Sets the text at the defined id.</td>
</tr>
<tr>
<td>void setSelectedById (int id)</td>
<td>Finds the specified entry by id, then marks its row as selected.</td>
</tr>
<tr>
<td>void setSelectedRow (int rowNum)</td>
<td></td>
</tr>
<tr>
<td>void sort (int columnId, bool increasing=true)</td>
<td>Performs a standard (alphabetical) sort on the values in the specified column.</td>
</tr>
<tr>
<td>void sortNumerical (int columnID, bool increasing=true)</td>
<td>Perform a numerical sort on the values in the specified column.</td>
</tr>
</tbody>
</table>

### Callbacks

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void onDeleteKey (string id)</td>
<td>Called when the delete key has been pressed.</td>
</tr>
<tr>
<td>void onSelect (string cellid, string text)</td>
<td>Called whenever an item in the list is selected.</td>
</tr>
</tbody>
</table>
## Public Attributes

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>bool</strong></td>
<td>clipColumnText</td>
<td>If true, text exceeding a column's given width will get clipped.</td>
</tr>
<tr>
<td><strong>intList</strong></td>
<td>columns</td>
<td>A vector of column offsets. The number of values determines the number of columns in the table.</td>
</tr>
<tr>
<td><strong>bool</strong></td>
<td>fitParentWidth</td>
<td>If true, the width of this control will match the width of its parent.</td>
</tr>
</tbody>
</table>
**Detailed Description**

GUI control that displays a list of text. Text items in the list can be individually selected.

**Example:**

```java
new GuiTextListCtrl(EndGameGuiList) {
    columns = "0 256";
    fitParentWidth = "1";
    clipColumnText = "0";
    //Properties not specific to this control have been omitted from this example.
};
```

**See also:**

Reference
Member Function Documentation

```cpp
int GuiTextListCtrl::addRow(int id = 0,
                          string text = "",
                          int index = -1)
```

Adds a new row at end of the list with the defined id and text. If index is used, then the new row is inserted at the row location of 'index'.

Parameters:

- `id` Id of the new row.
- `text` Text to display at the new row.
- `index` Index to insert the new row at. If not used, new row will be placed at the end of the list.

Example:

```cpp
// Define the id
%id = "4";

// Define the text to display
%text = "Display Text"

// Define the index (optional)
%index = "2"

// Inform the GuiTextListCtrl control to add a row
%rowIndex = %thisGuiTextListCtrl.addRow(%id, %text, %index)
```

Returns:

Returns the row index of the new row. If 'index' was defined, then this just returns the number of rows in the list.
See also:
References

```cpp
void GuiTextListCtrl::clear()
```
Clear the list.

**Example:**

```cpp
// Inform the GuiTextListCtrl control to clear its contents
%thisGuiTextListCtrl.clear();
```

See also:
GuiControl

```cpp
void GuiTextListCtrl::clearSelection()
```
Set the selection to nothing.

**Example:**

```cpp
// Deselect anything that is currently selected
%thisGuiTextListCtrl.clearSelection();
```

See also:
GuiControl

```cpp
int GuiTextListCtrl::findTextIndex(string needle )
```
Find needle in the list, and return the row number it was found in.

**Parameters:**

- `needle` Text to find in the list.
Example:

```cpp
// Define the text to find in the list
%needle = "Text To Find";

// Request the row number that contains the defined text
%rowNumber = %thisGuiTextListCtrl.findText(%needle);
```

Returns:
Row number that the defined text was found in,

See also:
GuiControl

```cpp
int GuiTextListCtrl::getRowId(int index)
```
Get the row ID for an index.

Parameters:

- `index` Index to get the RowID at

Example:

```cpp
// Define the index
%index = "3";

// Request the row ID at the defined index
%rowId = %thisGuiTextListCtrl.getRowId(%index);
```

Returns:
RowID at the defined index.

See also:
GuiControl
int GuiTextListCtrl::getRowNumById(int id)

Get the row number for a specified id.

Parameters:

  id  Id to get the row number at

Example:

```cpp
// Define the id
%id = "4";

// Request the row number from the GuiText
%rowNumber = %thisGuiTextListCtrl.getRowNumById(%id);
```

See also:

GuiControl

string GuiTextListCtrl::getRowText(int index)

Get the text of the row with the specified index.

Parameters:

  index  Row index to acquire the text at.

Example:

```cpp
// Define the row index
%index = "5";

// Request the text from the row at the defined index
%rowText = %thisGuiTextListCtrl.getRowText(%index);
```
Text at the defined row index.

**See also:**

`GuiControl`

```cpp
string GuiTextListCtrl::getRowTextById(int id)
```

Get the text of a row with the specified id.

**Example:**

```cpp
// Define the id
$id = "4";

// Inform the GuiTextListCtrl control to return the text at the defined row id
$rowText = $thisGuiTextListCtrl.getRowTextById($id);
```

**Returns:**

Row text at the requested row id.

**See also:**

`GuiControl`

```cpp
int GuiTextListCtrl::getSelectedId()
```

Get the ID of the currently selected item.

**Example:**

```cpp
// Acquire the ID of the selected item in the list
$id = $thisGuiTextListCtrl.getSelectedId();
```

**Returns:**

The id of the selected item in the list.
See also:
   GuiControl

int GuiTextListCtrl::getSelectedRow( )

Returns the selected row index (not the row ID).

Example:
   // Acquire the selected row index
   %rowIndex = %thisGuiTextListCtrl.getSelectedRow();

Returns:
   Index of the selected row

See also:
   GuiControl

bool GuiTextListCtrl::isRowActive(int rowNum )

Check if the specified row is currently active or not.

Parameters:
   rowNum  Row number to check the active state.

Example:
   // Define the row number
   %rowNum = "5";

   // Request the active state of the defined row
   %rowActiveState = %thisGuiTextListCtrl.isRowActive(%rowNum);

Returns:
Active state of the defined row number.

See also:
    GuiControl

void GuiTextListCtrl::onDeleteKey(string id)

Called when the delete key has been pressed.

Parameters:
    id  Id of the selected item in the list

Example:

    // The delete key was pressed while the GuiTextListCtrl was in focus,
    GuiTextListCtrl::onDeleteKey(%this,%id)
    {
        // Code to run when the delete key is pressed
    }

See also:
    GuiControl

void GuiTextListCtrl::onSelect(string cellid, string text)

Called whenever an item in the list is selected.

Parameters:
    cellid  The ID of the cell that was selected
    text    The text in the selected cell

Example:
// A cel in the control was selected, causing the callback to occur
GuiTextListCtrl::onSelect(%this,%callid,%text)
{
    // Code to run when a cel item is selected
}

See also:
GuiControl

void GuiTextListCtrl::removeRow(int index )

Remove a row from the table, based on its index.

Parameters:

index Row index to remove from the list.

Example:

    // Define the row index
    %index = "4";

    // Inform the GuiTextListCtrl control to remove the row
    %thisGuiTextListCtrl.removeRow(%index);

See also:
GuiControl

void GuiTextListCtrl::removeRowById(int id )

Remove row with the specified id.

Parameters:

id Id to remove the row entry at
Example:

```
// Define the id
%id = "4";

// Inform the GuiTextListCtrl control to:
%thisGuiTextListCtrl.removeRowById(%id);
```

See also:

GuiControl

---

**int GuiTextListCtrl::rowCount( )**

Get the number of rows.

**Example:**

```
// Get the number of rows in the list
%rowCount = %thisGuiTextListCtrl.rowCount();
```

**Returns:**

Number of rows in the list.

See also:

GuiControl

---

**void GuiTextListCtrl::scrollVisible (int rowNum )**

Scroll so the specified row is visible.

**Parameters:**

*rowNum*  Row number to make visible

**Example:**
// Define the row number to make visible
%rowNum = "4";

// Inform the GuiTextListCtrl control to set
%thisGuiTextListCtrl.scrollVisible(%rowNum);

See also:
GuiControl

void GuiTextListCtrl::setRowActive(int rowNum, bool active)

Mark a specified row as active/not.

Parameters:
rowNum  Row number to change the active state.
active    Boolean active state to set the row number.

Example:

// Define the row number
%rowNum = "4";

// Define the boolean active state
%active = "true";

// Inform the GuiTextListCtrl control to set
%thisGuiTextListCtrl.setRowActive(%rowNum,

See also:
GuiControl
void GuiTextListCtrl::setRowById(int id, string text)

Sets the text at the defined id.

**Parameters:**

- *id*  Id to change.
- *text*  Text to use at the Id.

**Example:**

```
// Define the id
%id = "4";

// Define the text
%text = "Text To Display";

// Inform the GuiTextListCtrl control to do
%thisGuiTextListCtrl.setRowById(%id,%text);
```

See also:

GuiControl

void GuiTextListCtrl::setSelectedById(int id)

Finds the specified entry by id, then marks its row as selected.

**Parameters:**

- *id*  Entry within the text list to make selected.

**Example:**

```
// Define the id
%id = "5";
```
// Inform the GuiTextListCtrl control to set
%thisGuiTextListCtrl.setSelectedById(%id);

See also:
GuiControl

void GuiTextListCtrl::setSelectedRow(int rowNum)

the specified row.

Parameters:

rowNum  Row number to set selected.

Example:

// Define the row number to set selected
%rowNum = "4";

%guiTextListCtrl.setSelectedRow(%rowNum);

See also:
GuiControl

void GuiTextListCtrl::sort(int  columnId,
    bool increasing = true
)

Performs a standard (alphabetical) sort on the values in the specified column.

Parameters:

columnId  Column ID to perform the sort on.

increasing  If false, sort will be performed in reverse.
Example:

```c
// Define the columnId
%id = "1";

// Define if we are increasing or not
%increasing = "false";

// Inform the GuiTextListCtrl to perform the sort operation
%thisGuiTextListCtrl.sort(%id,%increasing);
```

See also:

`GuiControl`

```c
void GuiTextListCtrl::sortNumerical(int  columnID, bool  increasing = true )
```

Perform a numerical sort on the values in the specified column.

**Detailed description**

**Parameters:**

- `columnId`  Column ID to perform the sort on.
- `increasing`  If false, sort will be performed in reverse.

**Example:**

```c
// Define the columnId
%id = "1";

// Define if we are increasing or not
%increasing = "false";

// Inform the GuiTextListCtrl to perform the sort operation
%thisGuiTextListCtrl.sort(%id,%increasing);
```
%thisGuiTextListCtrl.sortNumerical(%id,%ir

See also:
GuiControl
Member Data Documentation

bool GuiTextListCtrl::clipColumnText

If true, text exceeding a column's given width will get clipped.

intList GuiTextListCtrl::columns

A vector of column offsets. The number of values determines the number of columns in the table.

bool GuiTextListCtrl::fitParentWidth

If true, the width of this control will match the width of its parent.

Copyright © GarageGames, LLC. All Rights Reserved.
GuiTheoraCtrl Class Reference
[Image and Video Controls]

A control to playing Theora videos. More...

Inheritance diagram for GuiTheoraCtrl:

```
  SimObject
     |
     |
  SimSet
     |
     |
  SimGroup
     |
     |
GuiControl
     |
     |
GuiTheoraCtrl
```

List of all members.
# Public Member Functions

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>getcurrentTime()</code></td>
<td>Get the current playback time.</td>
</tr>
<tr>
<td>bool</td>
<td><code>isPlaybackDone()</code></td>
<td>Test whether the video has finished playing.</td>
</tr>
<tr>
<td>void</td>
<td><code>pause()</code></td>
<td>Pause playback of the video. If the video is not currently playing, the call is ignored.</td>
</tr>
<tr>
<td>void</td>
<td><code>play()</code></td>
<td>Start playing the video. If the video is already playing, the call is ignored.</td>
</tr>
<tr>
<td>void</td>
<td><code>setFile(string filename)</code></td>
<td>Set the video file to play. If a video is already playing, playback is stopped and the new video file is loaded.</td>
</tr>
<tr>
<td>void</td>
<td><code>stop()</code></td>
<td>Stop playback of the video. The next call to <code>play()</code> will then start playback from the beginning of the video.</td>
</tr>
</tbody>
</table>
## Public Attributes

### Playback

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Color</strong></td>
<td><code>backgroundColor</code> Fill color when video is not playing.</td>
</tr>
<tr>
<td><strong>bool</strong></td>
<td><code>matchVideoSize</code> Whether to automatically match control extents to the video size.</td>
</tr>
<tr>
<td><strong>bool</strong></td>
<td><code>playOnWake</code> Whether to start playing video when control is woken up.</td>
</tr>
<tr>
<td><strong>bool</strong></td>
<td><code>renderDebugInfo</code> If true, displays an overlay on top of the video with useful debugging information.</td>
</tr>
<tr>
<td><strong>bool</strong></td>
<td><code>stopOnSleep</code> Whether to stop video when control is set to sleep.</td>
</tr>
<tr>
<td><strong>filename</strong></td>
<td><code>theoraFile</code> Theora video file to play.</td>
</tr>
<tr>
<td><strong>GuiTheoraTranscoder</strong></td>
<td><code>transcoder</code> The routine to use for Y'CbCr to RGB conversion.</td>
</tr>
</tbody>
</table>
Detailed Description

A control to playing Theora videos.

This control can be used to play videos in the Theora video format. The videos may include audio in Vorbis format. The codecs for both formats are integrated with the engine and no codecs must be present on the user's machine.

Example:

```javascript
%video = new GuiTheoraCtrl()
{
    theoraFile = "videos/intro.ogv";
    playOnWake = false;
    stopOnSleep = true;
}

Canvas.setContent( %video );
%video.play();
```

See also:

http://www.theora.org
Member Function Documentation

float GuiTheoraCtrl::getCurrentTime( )

Get the current playback time.

**Returns:**

The elapsed playback time in seconds.

bool GuiTheoraCtrl::isPlaybackDone( )

Test whether the video has finished playing.

**Returns:**

True if the video has finished playing, false otherwise.

void GuiTheoraCtrl::pause( )

Pause playback of the video. If the video is not currently playing, the call is ignored.

While stopped, the control displays the last frame.

void GuiTheoraCtrl::play( )

Start playing the video. If the video is already playing, the call is ignored.

void GuiTheoraCtrl::setFile(string filename )

Set the video file to play. If a video is already playing, playback is
stopped and the new video file is loaded.

**Parameters:**

*filename* The video file to load.

```c
void GuiTheoraCtrl::stop()
```

Stop playback of the video. The next call to `play()` will then start playback from the beginning of the video.

While stopped, the control renders empty with just the background color.
# Member Data Documentation

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Color GuiTheoraCtrl::backgroundColor</code></td>
<td>Fill color when video is not playing.</td>
</tr>
<tr>
<td><code>bool GuiTheoraCtrl::matchVideoSize</code></td>
<td>Whether to automatically match control extents to the video size.</td>
</tr>
<tr>
<td><code>bool GuiTheoraCtrl::playOnWake</code></td>
<td>Whether to start playing video when control is woken up.</td>
</tr>
<tr>
<td><code>bool GuiTheoraCtrl::renderDebugInfo</code></td>
<td>If true, displays an overlay on top of the video with useful debugging information.</td>
</tr>
<tr>
<td><code>bool GuiTheoraCtrl::stopOnSleep</code></td>
<td>Whether to stop video when control is set to sleep.</td>
</tr>
<tr>
<td></td>
<td>If this is not set to true, the video will be paused when the control is put to sleep. This is because there is no support for seeking in the video stream in the player backend and letting the time source used to synchronize video (either audio or a raw timer) get far ahead of frame decoding will cause possibly very long delays when the control is woken up again.</td>
</tr>
<tr>
<td><code>filename GuiTheoraCtrl::theoraFile</code></td>
<td></td>
</tr>
</tbody>
</table>
Theora video file to play.

GuiTheoraTranscoder GuiTheoraCtrl::transcoder

The routine to use for Y’CbCr to RGB conversion.

Copyright © GarageGames, LLC. All Rights Reserved.
GuiTickCtrl Class Reference
[Utility Controls]

Brief Description. More...

Inheritance diagram for GuiTickCtrl:

```
        SimObject
         |
         V
        SimSet
         |
         V
        SimGroup
         |
         V
       GuiControl
         |
         V
      GuiTickCtrl
            |
       GuiAutoScrollCtrl
            |
      GuiMenuBar
```

List of all members.
Public Member Functions

void setProcessTicks (bool tick)

This will set this object to either be processing ticks or not.
Detailed Description

Brief Description.

This Gui Control is designed to be subclassed to let people create controls which want to receive update ticks at a constant interval. This class was created to be the Parent class of a control which used a DynamicTexture along with a VectorField to create warping effects much like the ones found in visualization displays for iTunes or Winamp. Those displays are updated at the framerate frequency. This works fine for those effects, however for an application of the same type of effects for things like Gui transitions the framerate-driven update frequency is not desirable because it does not allow the developer to be able to have any idea of a consistent user-experience.

Enter the ITickable interface. This lets the Gui control, in this case, update the dynamic texture at a constant rate of once per tick, even though it gets rendered every frame, thus creating a framerate-independent update frequency so that the effects are at a consistent speed regardless of the specifics of the system the user is on. This means that the screen-transitions will occur in the same time on a machine getting 300fps in the Gui shell as a machine which gets 150fps in the Gui shell.
Member Function Documentation

```c
void GuiTickCtrl::setProcessTicks(bool tick)
```

This will set this object to either be processing ticks or not.

Parameters:

\(\text{tick}\) (optional) True or nothing to enable ticking, false otherwise.

Example:

```c
// Turn off ticking for a control, like a
%sampleMenuBar.setProcessTicks(false);
```

Copyright © GarageGames, LLC. All Rights Reserved.
GuiToggleButtonCtrl Class Reference

[Button Controls]

Deprecated gui control. More...

Inheritance diagram for GuiToggleButtonCtrl:

```
          SimObject
             ↓
            SimSet
               ↓
              SimGroup
                 ↓
                GuiControl
                   ↓
                  GuiButtonBaseCtrl
                     ↓
                    GuiButtonCtrl
                       ↓
                      GuiToggleButtonCtrl
                      [legend]
```

List of all members.
Detailed Description
Deprecated gui control.

See also:
GuiButtonCtrl
GuiCheckBoxCtrl
GuiTreeViewCtrl Class Reference
[Container Controls]

Hierarchical list of text items with optional icons. More...

Inheritance diagram for GuiTreeViewCtrl:

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void addSelection (int id, bool isLastSelection=true)</code></td>
<td>Add an item/object to the current selection.</td>
</tr>
<tr>
<td><code>bool buildIconTable (builds an icon table)</code></td>
<td></td>
</tr>
<tr>
<td><code>void clear ()</code></td>
<td>empty tree</td>
</tr>
<tr>
<td><code>void clearFilterText ()</code></td>
<td>Clear the current item filtering pattern.</td>
</tr>
<tr>
<td><code>void clearSelection ()</code></td>
<td>Unselect all currently selected items.</td>
</tr>
<tr>
<td><code>void deleteSelection ()</code></td>
<td>Delete all items/objects in the current selection.</td>
</tr>
<tr>
<td><code>bool editItem (TreeItemId item, string newText, string newValue)</code></td>
<td></td>
</tr>
<tr>
<td><code>bool expandItem (TreeItemId item, bool expand=true)</code></td>
<td></td>
</tr>
<tr>
<td><code>int findChildItemByName (int parentId, string childName)</code></td>
<td>Get the child item of the given parent item whose text matches <code>childName</code>.</td>
</tr>
<tr>
<td><code>int findItemByName (string text)</code></td>
<td>Get the ID of the item whose text matches the given <code>text</code>.</td>
</tr>
<tr>
<td><code>int findItemByObjectId (find item by object id and returns the mId)</code></td>
<td></td>
</tr>
<tr>
<td><code>int findItemByValue (string value)</code></td>
<td>Get the ID of the item whose value matches <code>value</code>.</td>
</tr>
<tr>
<td><code>int getChild (TreeItemId item)</code></td>
<td></td>
</tr>
<tr>
<td><code>string getFilterText ()</code></td>
<td>Get the current filter expression. Only tree items whose text matches this expression are displayed. By default, the expression is empty and all items are shown.</td>
</tr>
<tr>
<td><code>string getItemText (TreeItemId item)</code></td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>getItemValue (TreeItemId item)</code></td>
<td>Return the selected item at the given index.</td>
</tr>
<tr>
<td><code>getNextSibling (TreeItemId item)</code></td>
<td></td>
</tr>
<tr>
<td><code>getParent (TreeItemId item)</code></td>
<td></td>
</tr>
<tr>
<td><code>getPrevSibling (TreeItemId item)</code></td>
<td></td>
</tr>
<tr>
<td><code>getSelectedIndex (int index=0)</code></td>
<td>Return the currently selected SimObject at the given index in inspector mode or -1.</td>
</tr>
<tr>
<td><code>getTextToRoot (TreeItemId item, Delimiter=none)</code></td>
<td>Gets the text from the current node to the root, concatenating at each branch upward, with a specified delimiter optionally.</td>
</tr>
<tr>
<td><code>hideSelection (bool state=true)</code></td>
<td>Call SimObject:: setHidden (state) on all objects in the current selection.</td>
</tr>
<tr>
<td><code>insertItem (int parentId, string text, string value=&quot;&quot;, string icon=&quot;&quot;, int normalImage=0, int expandedImage=0)</code></td>
<td>Add a new item to the tree.</td>
</tr>
<tr>
<td><code>isItemSelected (int id)</code></td>
<td>Check whether the given item is currently selected in the tree.</td>
</tr>
<tr>
<td><code>isParentItem (int id)</code></td>
<td>Returns true if the given item contains child items.</td>
</tr>
<tr>
<td><code>lockSelection (bool lock=true)</code></td>
<td>Set whether the current selection can be changed by the user or not.</td>
</tr>
<tr>
<td><code>markItem (TreeItemId item, bool mark=true)</code></td>
<td></td>
</tr>
<tr>
<td><code>moveItemDown (TreeItemId item)</code></td>
<td></td>
</tr>
<tr>
<td><code>moveItemUp (TreeItemId item)</code></td>
<td></td>
</tr>
<tr>
<td><code>open (SimSet obj, bool okToEdit=true)</code></td>
<td></td>
</tr>
</tbody>
</table>
Set the root of the tree view to the specified object, or to the root set.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool <code>removeItem</code> (TreeItemId item)</td>
<td>remove an item</td>
</tr>
<tr>
<td>void <code>removeSelection</code> ()</td>
<td>(deselects an item)</td>
</tr>
<tr>
<td>void <code>scrollVisible</code> (TreeItemId item)</td>
<td></td>
</tr>
<tr>
<td>int <code>scrollVisibleByObjectId</code> (show item by object id. returns true if successful.)</td>
<td></td>
</tr>
<tr>
<td>bool <code>selectItem</code> (TreeItemId item, bool select=true)</td>
<td>select an item</td>
</tr>
<tr>
<td>void <code>setDebug</code> (bool value=true)</td>
<td>Enable/disable debug output.</td>
</tr>
<tr>
<td>void <code>setFilterText</code> (string pattern)</td>
<td>Set the pattern by which to filter items in the tree. Only items in the tree whose text matches this pattern are displayed.</td>
</tr>
<tr>
<td>void <code>setItemImages</code> (int id, int normalImage, int expandedImage)</td>
<td>Sets the normal and expanded images to show for the given item.</td>
</tr>
<tr>
<td>void <code>setItemTooltip</code> (int id, string text)</td>
<td>Set the tooltip to show for the given item.</td>
</tr>
<tr>
<td>void <code>showItemRenameCtrl</code> (TreeItemId id)</td>
<td>Show the rename text field for the given item (only one at a time).</td>
</tr>
<tr>
<td>void <code>sort</code> (int parent, bool traverseHierarchy=false, bool parentsFirst=false, bool caseSensitive=true)</td>
<td>Sorts all items of the given parent (or root). With 'hierarchy', traverses hierarchy.</td>
</tr>
<tr>
<td>void <code>toggleHideSelection</code> ()</td>
<td>Toggle the hidden state of all objects in the current selection.</td>
</tr>
<tr>
<td>void <code>toggleLockSelection</code> ()</td>
<td>Toggle the locked state of all objects in the current selection.</td>
</tr>
</tbody>
</table>
## Callbacks

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bool canRenameObject (SimObject object)</code></td>
<td>Determines if an object can be renamed</td>
</tr>
<tr>
<td><code>bool handleRenameObject (string newName, SimObject object)</code></td>
<td>Handles the renaming of an object</td>
</tr>
<tr>
<td><code>bool isValidDragTarget (int id, string value)</code></td>
<td>Validates if an object is a valid drag target</td>
</tr>
<tr>
<td><code>void onAddGroupSelected (SimGroup group)</code></td>
<td>Called when a group is added to the selection</td>
</tr>
<tr>
<td><code>void onAddMultipleSelectionBegin ()</code></td>
<td>Called when multiple selections begin</td>
</tr>
<tr>
<td><code>void onAddMultipleSelectionEnd ()</code></td>
<td>Called when multiple selections end</td>
</tr>
<tr>
<td><code>void onAddSelection (int itemOrObjectId, bool isLastSelection)</code></td>
<td>Called when an item or object is added to the selection</td>
</tr>
<tr>
<td><code>void onBeginReparenting ()</code></td>
<td>Called when reparenting begins</td>
</tr>
<tr>
<td><code>void onClearSelection ()</code></td>
<td>Called when all selections are cleared</td>
</tr>
<tr>
<td><code>void onDefineIcons ()</code></td>
<td>Defines icons for items or objects</td>
</tr>
<tr>
<td><code>bool onDeleteObject (SimObject object)</code></td>
<td>Deletes an object</td>
</tr>
<tr>
<td><code>void onDeleteSelection ()</code></td>
<td>Called when an item or object is deleted</td>
</tr>
<tr>
<td><code>void onDragDropped ()</code></td>
<td>Called when dragging is dropped</td>
</tr>
<tr>
<td><code>void onEndReparenting ()</code></td>
<td>Called when reparenting ends</td>
</tr>
<tr>
<td><code>void onInspect (int itemOrObjectId)</code></td>
<td>Called when an item or object is inspected</td>
</tr>
<tr>
<td><code>void onKeyDown (int modifier, int keyCode)</code></td>
<td>Called when a key is pressed</td>
</tr>
<tr>
<td><code>void onMouseDragged ()</code></td>
<td>Called when the mouse is dragged</td>
</tr>
<tr>
<td><code>void onMouseUp (int hitItemId, int mouseClickCount)</code></td>
<td>Called when the mouse button is released</td>
</tr>
<tr>
<td><code>void onObjectDeleteCompleted ()</code></td>
<td>Called when an object is deleted</td>
</tr>
<tr>
<td><code>void onRemoveSelection (int itemOrObjectId)</code></td>
<td>Called when an item or object is removed</td>
</tr>
<tr>
<td><code>void onReparent (int itemOrObjectId, int oldParentItemOrObjectId, int newParentItemOrObjectId)</code></td>
<td>Reparents an item or object</td>
</tr>
<tr>
<td><code>void onRightMouseDown (int itemId, Point2I mousePos, SimObject object)</code></td>
<td>Called when the right mouse button is pressed</td>
</tr>
<tr>
<td><code>void onRightMouseUp (int itemId, Point2I mousePos, SimObject object)</code></td>
<td>Called when the right mouse button is released</td>
</tr>
<tr>
<td><code>void onSelect (int itemOrObjectId)</code></td>
<td>Called when an item or object is selected</td>
</tr>
</tbody>
</table>
void onUnselect (int itemOrObjectId)
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Method Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>addChildSelectionByValue</td>
<td><code>addChildSelectionByValue(TreeItemId parent, value)</code></td>
</tr>
<tr>
<td>void</td>
<td>buildVisibleTree</td>
<td>Build the visible tree.</td>
</tr>
<tr>
<td>void</td>
<td>cancelRename</td>
<td>For internal use.</td>
</tr>
<tr>
<td>int</td>
<td>getFirstRootItem</td>
<td>Get id for root item.</td>
</tr>
<tr>
<td>int</td>
<td>getItemCount</td>
<td></td>
</tr>
<tr>
<td>string</td>
<td>getSelectedItemList</td>
<td><code>getSelectedItemList</code> returns a space seperated list of multiple item ids</td>
</tr>
<tr>
<td>int</td>
<td>getSelectedItemsCount</td>
<td></td>
</tr>
<tr>
<td>string</td>
<td>getSelectedObjectList</td>
<td><code>getSelectedObjectList</code> Returns a space sperated list of all selected object ids.</td>
</tr>
<tr>
<td>void</td>
<td>onRenameValidate</td>
<td>For internal use.</td>
</tr>
<tr>
<td>void</td>
<td>removeAllChildren</td>
<td><code>removeAllChildren(TreeItemId parent)</code></td>
</tr>
<tr>
<td>void</td>
<td>removeChildSelectionByValue</td>
<td><code>removeChildSelectionByValue(TreeItemId parent, value)</code></td>
</tr>
</tbody>
</table>

## Inspector Trees

<table>
<thead>
<tr>
<th>Type</th>
<th>Method Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>canRenameObjects</td>
<td>If true clicking on a selected item ( that is an object and not the root ) will allow you to rename it.</td>
</tr>
<tr>
<td>bool</td>
<td>compareToObjectID</td>
<td></td>
</tr>
<tr>
<td>bool</td>
<td>renameInternal</td>
<td>If true then object renaming operates on the internalName</td>
</tr>
</tbody>
</table>
rather than the object name.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool showClassNameForUnnamedObjects</td>
<td>If true, class names will be used as object names for unnamed objects.</td>
</tr>
<tr>
<td>bool showClassNames</td>
<td>If true, item text labels for objects will include class names.</td>
</tr>
<tr>
<td>bool showInternalNames</td>
<td>If true, item text labels for objects will include internal names.</td>
</tr>
<tr>
<td>bool showObjectIds</td>
<td>If true, item text labels for objects will include object IDs.</td>
</tr>
<tr>
<td>bool showObjectNames</td>
<td>If true, item text labels for objects will include object names.</td>
</tr>
</tbody>
</table>

**TreeView**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool clearAllOnSingleSelection</td>
<td></td>
</tr>
<tr>
<td>bool deleteObjectAllowed</td>
<td></td>
</tr>
<tr>
<td>bool destroyTreeOnSleep</td>
<td>If true, the entire tree item hierarchy is deleted when the control goes to sleep.</td>
</tr>
<tr>
<td>bool dragToItemAllowed</td>
<td></td>
</tr>
<tr>
<td>bool fullRowSelect</td>
<td></td>
</tr>
<tr>
<td>int itemHeight</td>
<td></td>
</tr>
<tr>
<td>bool mouseDragging</td>
<td></td>
</tr>
<tr>
<td>bool multipleSelections</td>
<td>If true, multiple items can be selected concurrently.</td>
</tr>
<tr>
<td>bool showRoot</td>
<td>If true, the root item is shown in the tree.</td>
</tr>
<tr>
<td>int tabSize</td>
<td></td>
</tr>
<tr>
<td>int textOffset</td>
<td></td>
</tr>
<tr>
<td>bool</td>
<td>tooltipOnWidthOnly</td>
</tr>
<tr>
<td>-------</td>
<td>--------------------</td>
</tr>
<tr>
<td>bool</td>
<td>useInspectorToolips</td>
</tr>
</tbody>
</table>
Detailed Description

Hierarchical list of text items with optional icons.

Can also be used to inspect SimObject hierarchies, primarily within editors.

GuiTreeViewCtrls can either display arbitrary user-defined trees or can be used to display SimObject hierarchies where each parent node in the tree is a SimSet or SimGroup and each leaf node is a SimObject.

Each item in the tree has a text and a value. For trees that display SimObject hierarchies, the text for each item is automatically derived from objects while the value for each item is the ID of the respective SimObject. For trees that are not tied to SimObjects, both text and value of each item are set by the user.

Additionally, items in the tree can have icons.

Each item in the tree has a distinct numeric ID that is unique within its tree. The ID of the root item, which is always present on a tree, is 0.

Example:

```c
new GuiTreeViewCtrl(DatablockEditorTree) {
    tabSize = "16";
    textOffset = "2";
    fullRowSelect = "0";
    itemHeight = "21";
    destroyTreeOnSleep = "0";
    MouseDragging = "0";
    MultipleSelections = "1";
    DeleteObjectAllowed = "1";
    DragToItemAllowed = "0";
}```
ClearAllOnSingleSelection = "1";
showRoot = "1";
internalNamesOnly = "0";
objectNamesOnly = "0";
compareToObjectID = "0";
Profile = "GuiTreeViewProfile"
tooltipprofile = "GuiToolTipProfile"
hovertime = "1000";
}
Member Function Documentation

void GuiTreeViewCtrl::addSelection(int id, bool isLastSelection = true)

Add an item/object to the current selection.

Parameters:

- **id**: ID of item/object to add to the selection.
- **isLastSelection**: Whether there are more pending items/objects to be added to the selection. If false, the control will defer refreshing the tree and wait until addSelection() is called with this parameter set to true.

bool GuiTreeViewCtrl::buildIconTable()

builds an icon table

bool GuiTreeViewCtrl::canRenameObject(SimObject object)

void GuiTreeViewCtrl::clear()

empty tree

void GuiTreeViewCtrl::clearFilterText()

Clear the current item filtering pattern.

See also:

setFilterText
getFilterText
void GuiTreeCtrl::clearSelection()  
Unselect all currently selected items.

void GuiTreeCtrl::deleteSelection()  
Delete all items/objects in the current selection.

bool GuiTreeCtrl::editItem(TreeItemId item,  
    string newText,  
    string newValue  
)

bool GuiTreeCtrl::expandItem(TreeItemId item,  
    bool expand = true  
)

int GuiTreeCtrl::findChildItemByName(int parentId,  
    string childName  
)

Get the child item of the given parent item whose text matches childName.

Parameters:

    parentId Item ID of the parent in which to look for the child.
    childName Text of the child item to find.

Returns:

    ID of the child item or -1 if no child in parentId has the given text childName.

Note:
This method does not recurse, i.e. it only looks for direct children.

```cpp
int GuiTreeViewCtrl::findItemByName(string text)
```

Get the ID of the item whose text matches the given `text`.

**Parameters:**
- `text` Item text to match.

**Returns:**
- ID of the item or -1 if no item matches the given text.

```cpp
int GuiTreeViewCtrl::findItemByObjectId(find item by object id and retu
```

```cpp
int GuiTreeViewCtrl::findItemByValue(string value)
```

Get the ID of the item whose value matches `value`.

**Parameters:**
- `value` Value text to match.

**Returns:**
- ID of the item or -1 if no item has the given value.

```cpp
int GuiTreeViewCtrl::getChild(TreeItemId item)
```

```cpp
string GuiTreeViewCtrl::getFilterText()
```

Get the current filter expression. Only tree items whose text matches this expression are displayed. By default, the expression is empty and all items are shown.
**Returns:**

The current filter pattern or an empty string if no filter pattern is currently active.

**See also:**
- `setFilterText`
- `clearFilterText`

```
string GuiTreeViewCtrl::getItemText(TreeItemId item )

string GuiTreeViewCtrl::getItemValue(TreeItemId item )

int GuiTreeViewCtrl::getNextSibling(TreeItemId item )

int GuiTreeViewCtrl::getParent(TreeItemId item )

int GuiTreeViewCtrl::getPrevSibling(TreeItemId item )

int GuiTreeViewCtrl::getSelectedItem(int index = 0 )

Return the selected item at the given index.

int GuiTreeViewCtrl::getSelectedObject(int index = 0 )

Return the currently selected `SimObject` at the given index in inspector mode or -1.

string GuiTreeViewCtrl::getTextToRoot(TreeItemId item, Delimiter = none )
```
gets the text from the current node to the root, concatenating at each branch upward, with a specified delimiter optionally

```cpp
bool GuiTreeViewCtrl::handleRenameObject(string newName, SimObject object)
```

```cpp
void GuiTreeViewCtrl::hideSelection(bool state = true)
```

Call `SimObject::setHidden(state)` on all objects in the current selection.

**Parameters:**

- `state` Visibility state to set objects in selection to.

```cpp
int GuiTreeViewCtrl::insertItem(int parentId, string text, string value = "", string icon = "", int normalImage = 0, int expandedImage = 0)
```

Add a new item to the tree.

**Parameters:**

- `parentId` Item ID of parent to which to add the item as a child. 0 is root item.
- `text` Text to display on the item in the tree.
- `value` Behind-the-scenes value of the item.
**Returns:**
The ID of the newly added item.

```cpp
bool GuiTreeCtrl::isItemSelected (int id)
```
Check whether the given item is currently selected in the tree.

**Parameters:**

- `id` Item/object ID.

**Returns:**
True if the given item/object is currently selected in the tree.

```cpp
bool GuiTreeCtrl::isParentItem (int id)
```
Returns true if the given item contains child items.

```cpp
bool GuiTreeCtrl::isValidDragTarget (int id, string value)
```

```cpp
void GuiTreeCtrl::lockSelection (bool lock = true)
```
Set whether the current selection can be changed by the user or not.

**Parameters:**

- `lock` If true, the current selection is frozen and cannot be changed. If false, the selection may be modified.

```cpp
bool GuiTreeCtrl::markItem (TreeItemId item, int id)
```
bool
mark = true
)

void GuiTreeCtrl::moveItemDown(TreeItemId item)

void GuiTreeCtrl::moveItemUp(TreeItemId item)

void GuiTreeCtrl::onAddGroupSelected(SimGroup group)

void GuiTreeCtrl::onAddMultipleSelectionBegin()

void GuiTreeCtrl::onAddMultipleSelectionEnd()

void GuiTreeCtrl::onAddSelection(int itemOrObjectId, bool isLastSelection)

void GuiTreeCtrl::onBeginReparenting()

void GuiTreeCtrl::onClearSelection()

void GuiTreeCtrl::onDefineIcons()

bool GuiTreeCtrl::onDeleteObject(SimObject object)

void GuiTreeCtrl::onDeleteSelection()

void GuiTreeCtrl::onDragDropped()
<table>
<thead>
<tr>
<th>Function</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>GuiTreeViewCtrl::onEndReparenting</td>
<td>()</td>
</tr>
<tr>
<td>GuiTreeViewCtrl::onInspect</td>
<td>(int itemOrObjectId)</td>
</tr>
<tr>
<td>GuiTreeViewCtrl::onKeyDown</td>
<td>(int modifier, int keyCode)</td>
</tr>
<tr>
<td>GuiTreeViewCtrl::onMouseDragged</td>
<td>()</td>
</tr>
<tr>
<td>GuiTreeViewCtrl::onMouseUp</td>
<td>(int hitItemId, int mouseClickedCount)</td>
</tr>
<tr>
<td>GuiTreeViewCtrl::onObjectDeleteCompleted</td>
<td>()</td>
</tr>
<tr>
<td>GuiTreeViewCtrl::onRemoveSelection</td>
<td>(int itemOrObjectId)</td>
</tr>
<tr>
<td>GuiTreeViewCtrl::onReparent</td>
<td>(int itemOrObjectId, int oldParentItemOrObjectId, int newParentItemOrObjectId)</td>
</tr>
<tr>
<td>GuiTreeViewCtrl::onRightMouseDown</td>
<td>(int itemId, Point2I mousePos, SimObject object)</td>
</tr>
<tr>
<td>GuiTreeViewCtrl::onRightMouseUp</td>
<td>(int itemId, Point2I mousePos, SimObject object)</td>
</tr>
</tbody>
</table>
### GuiTreeViewCtrl Methods

#### onSelect

```cpp
void GuiTreeViewCtrl::onSelect(int itemOrObjectId)
```

#### onUnselect

```cpp
void GuiTreeViewCtrl::onUnselect(int itemOrObjectId)
```

#### open

```cpp
void GuiTreeViewCtrl::open(SimSet obj,
                          bool okToEdit = true)
```

Set the root of the tree view to the specified object, or to the root set.

#### removeItem

```cpp
bool GuiTreeViewCtrl::removeItem(TreeItemId item)
```

#### removeSelection

```cpp
void GuiTreeViewCtrl::removeSelection(deselects an item)
```

#### scrollVisible

```cpp
void GuiTreeViewCtrl::scrollVisible(TreeItemId item)
```

#### scrollVisibleByObjectId

```cpp
int GuiTreeViewCtrl::scrollVisibleByObjectId(show item by object id.
  returns true if
```

#### selectItem

```cpp
bool GuiTreeViewCtrl::selectItem(TreeItemId item,
                                 bool select = true)
```

#### setDebug

```cpp
void GuiTreeViewCtrl::setDebug(bool value = true)
```

Enable/disable debug output.

#### setFilterText

```cpp
void GuiTreeViewCtrl::setFilterText(string pattern)
```
Set the pattern by which to filter items in the tree. Only items in the tree whose text matches this pattern are displayed.

**Parameters:**

- `pattern`: New pattern based on which visible items in the tree should be filtered. If empty, all items become visible.

**See also:**

- `getFilterText`
- `clearFilterText`

```c++
void GuiTreeViewCtrl::setItemImages(int id,
    int normallImage,
    int expandedImage
)
```

Sets the normal and expanded images to show for the given item.

```c++
void GuiTreeViewCtrl::setItemTooltip(int id,
    string text
)
```

Set the tooltip to show for the given item.

```c++
void GuiTreeViewCtrl::showItemRenameCtrl(TreeItemId id)
```

Show the rename text field for the given item (only one at a time).

```c++
void GuiTreeViewCtrl::sort(int parent,
    bool traverseHierarchy = false,
    bool parentsFirst = false,
```
bool caseSensitive = true
)

Sorts all items of the given parent (or root). With 'hierarchy', traverses hierarchy.

void GuiTreeViewCtrl::toggleHideSelection( )

Toggle the hidden state of all objects in the current selection.

void GuiTreeViewCtrl::toggleLockSelection( )

Toggle the locked state of all objects in the current selection.
Member Data Documentation

void GuiTreeCtrl::addChildSelectionByValue

addChildSelectionByValue(TreeItemId parent, value)

void GuiTreeCtrl::buildVisibleTree

Build the visible tree.

void GuiTreeCtrl::cancelRename

For internal use.

bool GuiTreeCtrl::canRenameObjects

If true clicking on a selected item (that is an object and not the root) will allow you to rename it.

bool GuiTreeCtrl::clearAllOnSingleSelection

bool GuiTreeCtrl::compareToObjectId

bool GuiTreeCtrl::deleteObjectAllowed

bool GuiTreeCtrl::destroyTreeOnSleep

If true, the entire tree item hierarchy is deleted when the control goes to sleep.
bool GuiTreeViewCtrl::dragToItemAllowed

bool GuiTreeViewCtrl::fullRowSelect

int GuiTreeViewCtrl::getFirstRootItem

Get id for root item.

int GuiTreeViewCtrl::getItemCount

string GuiTreeViewCtrl::getSelectedItemList

returns a space seperated list of mulitple item ids

int GuiTreeViewCtrl::getSelectedItemsCount

string GuiTreeViewCtrl::getSelectedObjectList

Returns a space sperated list of all selected object ids.

int GuiTreeViewCtrl::itemHeight

bool GuiTreeViewCtrl::mouseDragging

bool GuiTreeViewCtrl::multipleSelections

If true, multiple items can be selected concurrently.
void GuiTreeViewCtrl::onRenameValidate

For internal use.

void GuiTreeViewCtrl::removeAllChildren

removeAllChildren(TreeItemId parent)

void GuiTreeViewCtrl::removeChildSelectionByValue

removeChildSelectionByValue(TreeItemId parent, value)

bool GuiTreeViewCtrl::renameInternal

If true then object renaming operates on the internalName rather than the object name.

bool GuiTreeViewCtrl::showClassNameForUnnamedObjects

If true, class names will be used as object names for unnamed objects.

bool GuiTreeViewCtrl::showClassNames

If true, item text labels for objects will include class names.

bool GuiTreeViewCtrl::showInternalNames

If true, item text labels for objects will include internal names.
bool GuiTreeCtrl::showObjectIds

If true, item text labels for objects will include object IDs.

bool GuiTreeCtrl::showObjectNames

If true, item text labels for objects will include object names.

bool GuiTreeCtrl::showRoot

If true, the root item is shown in the tree.

int GuiTreeCtrl::tabSize

int GuiTreeCtrl::textOffset

bool GuiTreeCtrl::tooltipOnWidthOnly

bool GuiTreeCtrl::useInspectorToolTips
GuiTSCtrl Class Reference
[3D Controls]

Abstract base class for controls that render 3D scenes. More...

Inheritance diagram for GuiTSCtrl:

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Function Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><strong>calculateViewDistance</strong> (float radius)</td>
<td>Given the camera's current FOV, get the distance from the camera's viewpoint at which the given radius will fit in the render area.</td>
</tr>
<tr>
<td>Point2F</td>
<td><strong>getWorldToScreenScale</strong> ()</td>
<td>Get the ratio between world-space units and pixels.</td>
</tr>
<tr>
<td>Point3F</td>
<td><strong>project</strong> (Point3F worldPosition)</td>
<td>Transform world-space coordinates to screen-space (x, y, depth) coordinates.</td>
</tr>
<tr>
<td>Point3F</td>
<td><strong>unproject</strong> (Point3F screenPosition)</td>
<td>Transform 3D screen-space coordinates (x, y, depth) to world space.</td>
</tr>
</tbody>
</table>
Public Attributes

Camera

<table>
<thead>
<tr>
<th>float</th>
<th>cameraZRot</th>
<th>Z rotation angle of camera.</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>forceFOV</td>
<td>The vertical field of view in degrees or zero to use the normal camera FOV.</td>
</tr>
</tbody>
</table>

Rendering

| float  | reflectPriority | The share of the per-frame reflection update work this control's rendering should run. |
Detailed Description

Abstract base class for controls that render 3D scenes.

GuiTSCtrl is the base class for controls that render 3D camera views in Torque. The class itself does not implement a concrete scene rendering. Use GuiObjectView to display individual shapes in the Gui and GameTSCtrl to render full scenes.

See also:

GameTSCtrl
GuiObjectView
Member Function Documentation

float GuiTSCtrl::calculateViewDistance(float radius)

Given the camera's current FOV, get the distance from the camera's viewpoint at which the given radius will fit in the render area.

**Parameters:**

radius
Radius in world-space units which should fit in the view.

**Returns:**

The distance from the viewpoint at which the given radius would be fully visible.

Point2F GuiTSCtrl::getWorldToScreenScale()

Get the ratio between world-space units and pixels.

**Returns:**

The amount of world-space units covered by the extent of a single pixel.

Point3F GuiTSCtrl::project(Point3F worldPosition)

Transform world-space coordinates to screen-space (x, y, depth) coordinates.

**Parameters:**

worldPosition
The world-space position to transform to screen-space.

**Returns:**
The

Point3F GuiTSCtrl::unproject(Point3F screenPosition)

Transform 3D screen-space coordinates (x, y, depth) to world space.

This method can be, for example, used to find the world-space position relating to the current mouse cursor position.

Parameters:

`screenPosition` The x/y position on the screen plus the depth from the screen-plane outwards.

Returns:

The world-space position corresponding to the given screen-space coordinates.
Member Data Documentation

float GuiTSCtrl::cameraZRot

Z rotation angle of camera.

float GuiTSCtrl::forceFOV

The vertical field of view in degrees or zero to use the normal camera FOV.

float GuiTSCtrl::reflectPriority

The share of the per-frame reflection update work this control's rendering should run.

The reflect update priorities of all visible GuiTSCtrls are added together and each control is assigned a share of the per-frame reflection update time according to its percentage of the total priority value.
GuiWindowCtrl Class Reference
[Container Controls]

A window with a title bar and an optional set of buttons. More...

Inheritance diagram for GuiWindowCtrl:

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>attachTo (GuiWindowCtrl window)</code></td>
<td>Bring the window to the front.</td>
</tr>
<tr>
<td><code>selectWindow ()</code></td>
<td>Set the window's collapsing state.</td>
</tr>
<tr>
<td><code>setCollapseGroup (bool state)</code></td>
<td>Toggle the window collapsing.</td>
</tr>
</tbody>
</table>

### Callbacks

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>onClose ()</code></td>
<td>Called when the close button has been pressed.</td>
</tr>
<tr>
<td><code>onCollapse ()</code></td>
<td>Called when the window is collapsed by clicking its title bar.</td>
</tr>
<tr>
<td><code>onMaximize ()</code></td>
<td>Called when the window has been maximized.</td>
</tr>
<tr>
<td><code>onMinimize ()</code></td>
<td>Called when the window has been minimized.</td>
</tr>
<tr>
<td><code>onRestore ()</code></td>
<td>Called when the window is restored from minimized, maximized, or collapsed state.</td>
</tr>
</tbody>
</table>
Static Public Member Functions

```
static void attach (GuiWindowCtrl bottomWindow, GuiWindowCtrl topWindow)

Attach bottomWindow to so that bottomWindow moves along with topWindow when it is dragged.
```
## Public Attributes

### Window

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>canClose</td>
<td>Whether the window has a close button.</td>
</tr>
<tr>
<td>bool</td>
<td>canCollapse</td>
<td>Whether the window can be collapsed by clicking its title bar.</td>
</tr>
<tr>
<td>bool</td>
<td>canMaximize</td>
<td>Whether the window has a maximize button.</td>
</tr>
<tr>
<td>bool</td>
<td>canMinimize</td>
<td>Whether the window has a minimize button.</td>
</tr>
<tr>
<td>bool</td>
<td>canMove</td>
<td>Whether the window can be moved by dragging its titlebar.</td>
</tr>
<tr>
<td>string</td>
<td>closeCommand</td>
<td>Script code to execute when the window is closed.</td>
</tr>
<tr>
<td>bool</td>
<td>edgeSnap</td>
<td>If true, the window will snap to the edges of other windows when moved close to them.</td>
</tr>
<tr>
<td>bool</td>
<td>resizeHeight</td>
<td>Whether the window can be resized vertically.</td>
</tr>
<tr>
<td>bool</td>
<td>resizeWidth</td>
<td>Whether the window can be resized horizontally.</td>
</tr>
<tr>
<td>string</td>
<td>text</td>
<td>Text label to display in titlebar.</td>
</tr>
</tbody>
</table>
Detailed Description

A window with a title bar and an optional set of buttons.

The GuiWindowCtrl class implements windows that can be freely placed within the render window. Additionally, the windows can be resized and maximized/minimized.

Example:

```cpp
new GuiWindowCtrl( MyWindow )
{
    text = "My Window"; // The text that is displayed on the title bar.
    resizeWidth = true; // Allow horizontal resizing by user via mouse.
    resizeHeight = true; // Allow vertical resizing by user via mouse.
    canClose = true; // Display a close button in the title bar.
    canMinimize = true; // Display a minimize button in the title bar.
    canMaximize = true; // Display a maximize button in the title bar.
};
```
Member Function Documentation

static void GuiWindowCtrl::attach(GuiWindowCtrl bottomWindow, GuiWindowCtrl topWindow)

Attach bottomWindow to so that bottomWindow moves along with topWindow when it is dragged.

Parameters:
  - bottomWindow
  - topWindow

void GuiWindowCtrl::attachTo(GuiWindowCtrl window)

void GuiWindowCtrl::onClose()

Called when the close button has been pressed.

void GuiWindowCtrl::onCollapse()

Called when the window is collapsed by clicking its title bar.

void GuiWindowCtrl::onMaximize()

Called when the window has been maximized.

void GuiWindowCtrl::onMinimize()

Called when the window has been minimized.
void GuiWindowCtrl::onRestore()

Called when the window is restored from minimized, maximized, or collapsed state.

void GuiWindowCtrl::selectWindow()

Bring the window to the front.

void GuiWindowCtrl::setCollapseGroup(bool state)

Set the window's collapsing state.

void GuiWindowCtrl::toggleCollapseGroup()

Toggle the window collapsing.
### Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>GuiWindowCtrl::canClose</td>
<td>Whether the window has a close button.</td>
</tr>
<tr>
<td>bool</td>
<td>GuiWindowCtrl::canCollapse</td>
<td>Whether the window can be collapsed by clicking its title bar.</td>
</tr>
<tr>
<td>bool</td>
<td>GuiWindowCtrl::canMaximize</td>
<td>Whether the window has a maximize button.</td>
</tr>
<tr>
<td>bool</td>
<td>GuiWindowCtrl::canMinimize</td>
<td>Whether the window has a minimize button.</td>
</tr>
<tr>
<td>bool</td>
<td>GuiWindowCtrl::canMove</td>
<td>Whether the window can be moved by dragging its titlebar.</td>
</tr>
<tr>
<td>string</td>
<td>GuiWindowCtrl::closeCommand</td>
<td>Script code to execute when the window is closed.</td>
</tr>
<tr>
<td>bool</td>
<td>GuiWindowCtrl::edgeSnap</td>
<td>If true, the window will snap to the edges of other windows when</td>
</tr>
</tbody>
</table>
moved close to them.

`bool GuiWindowCtrl::resizeHeight`

Whether the window can be resized vertically.

`bool GuiWindowCtrl::resizeWidth`

Whether the window can be resized horizontally.

`string GuiWindowCtrl::text`

Text label to display in titlebar.
HoverVehicle Class Reference
[ Vehicles ]

A hovering vehicle. More...

Inheritance diagram for HoverVehicle:

```
    SimObject
     |   
     v   
    NetObject
     |   
     v   
SceneObject
     |   
     v   
GameBase
     |   
     v   
ShapeBase
     |   
     v   
Vehicle
     |   
    HoverVehicle
```

List of all members.
**Static Public Attributes**

<table>
<thead>
<tr>
<th>static bool</th>
<th>isRenderable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static bool</th>
<th>isSelectable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

A hovering vehicle.

A hover vehicle is a vehicle that maintains a specific distance between the vehicle and the ground at all times; unlike a flying vehicle which is free to ascend and descend at will. The model used for the HoverVehicle has the following requirements:

Collision mesh

- A convex collision mesh at detail size -1.

JetNozzle0-1 nodes

- Particle emitter nodes used when thrusting forward.

JetNozzle2-3 nodes

- Particle emitter nodes used when thrusting downward.

JetNozzleX node

- Particle emitter node used when thrusting backward.

activateBack animation

- Non-cyclic animation sequence played when the vehicle begins thrusting forwards.

maintainBack animation

- Cyclic animation sequence played after activateBack when the vehicle continues thrusting forwards.
HoverVehicleData Class Reference

[ Vehicles ]

Defines the properties of a HoverVehicle. More...

Inheritance diagram for HoverVehicleData:

SimObject

SimDataBlock

GameBaseData

ShapeBaseData

VehicleData

HoverVehicleData

[legend]

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>brakingActivationSpeed</td>
<td>Maximum speed below which a braking force is applied.</td>
</tr>
<tr>
<td>float</td>
<td>brakingForce</td>
<td>Force generated by braking.</td>
</tr>
<tr>
<td>float</td>
<td>dragForce</td>
<td>Drag force factor that acts opposite to the vehicle velocity.</td>
</tr>
<tr>
<td>ParticleEmitterData</td>
<td>dustTrailEmitter</td>
<td>Emitter to generate particles for the vehicle's dust trail.</td>
</tr>
<tr>
<td>float</td>
<td>dustTrailFreqMod</td>
<td>Number of dust trail particles to generate based on vehicle speed.</td>
</tr>
<tr>
<td>Point3F</td>
<td>dustTrailOffset</td>
<td>&quot;X Y Z&quot; offset from the vehicle's origin from which to generate dust trail particles.</td>
</tr>
<tr>
<td>SFXProfile</td>
<td>engineSound</td>
<td>Looping engine sound.</td>
</tr>
<tr>
<td>float</td>
<td>floatingGravMag</td>
<td>Scale factor applied to the vehicle gravitational force when the vehicle is floating.</td>
</tr>
<tr>
<td>float</td>
<td>floatingThrustFactor</td>
<td>Scalar applied to the vehicle's thrust force when the vehicle is floating.</td>
</tr>
<tr>
<td>SFXProfile</td>
<td>floatSound</td>
<td>Looping sound played while the vehicle is floating.</td>
</tr>
<tr>
<td>ParticleEmitterData</td>
<td>forwardJetEmitter</td>
<td>Emitter to generate particles for forward jet.</td>
</tr>
</tbody>
</table>
thrust.

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>gyroDrag</td>
<td>Damping torque that acts against the vehicle's current angular momentum.</td>
</tr>
<tr>
<td></td>
<td>jetSound</td>
<td>Looping sound played when the vehicle is jetting.</td>
</tr>
<tr>
<td>float</td>
<td>mainThrustForce</td>
<td>Force generated by thrusting the vehicle forward.</td>
</tr>
<tr>
<td>float</td>
<td>normalForce</td>
<td>Force generated in the ground normal direction when the vehicle is not floating (within stabilizer length from the ground).</td>
</tr>
<tr>
<td>float</td>
<td>pitchForce</td>
<td>Pitch (rotation about the X-axis) force applied when steering in the y-axis direction.</td>
</tr>
<tr>
<td>float</td>
<td>restorativeForce</td>
<td>Force generated to stabilize the vehicle (return it to neutral pitch/roll) when the vehicle is floating (more than stabilizer length from the ground).</td>
</tr>
<tr>
<td>float</td>
<td>reverseThrustForce</td>
<td>Force generated by thrusting the vehicle backward.</td>
</tr>
<tr>
<td>float</td>
<td>rollForce</td>
<td>Roll (rotation about the Y-axis) force applied when steering in the x-axis direction.</td>
</tr>
<tr>
<td>float</td>
<td>stabDampingConstant</td>
<td>Damping spring force acting against changes in the stabilizer length.</td>
</tr>
<tr>
<td>float</td>
<td>stabLenMax</td>
<td></td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><code>stabLenMin</code></td>
<td>Length of the base stabilizer when travelling at maximum speed (maxThrustSpeed).</td>
<td></td>
</tr>
<tr>
<td><code>stabSpringConstant</code></td>
<td>Value used to generate stabilizer spring force. The force generated depends on stabilizer compression, that is how close the vehicle is to the ground proportional to current stabilizer length.</td>
<td></td>
</tr>
<tr>
<td><code>steeringForce</code></td>
<td>Yaw (rotation about the Z-axis) force applied when steering in the x-axis direction about the vehicle's Z-axis.</td>
<td></td>
</tr>
<tr>
<td><code>strafeThrustForce</code></td>
<td>Force generated by thrusting the vehicle to one side.</td>
<td></td>
</tr>
<tr>
<td><code>triggerTrailHeight</code></td>
<td>Maximum height above surface to emit dust trail particles.</td>
<td></td>
</tr>
<tr>
<td><code>turboFactor</code></td>
<td>Scale factor applied to the vehicle's thrust force when jetting.</td>
<td></td>
</tr>
<tr>
<td><code>vertFactor</code></td>
<td>Scalar applied to the vertical portion of the velocity drag acting on the vehicle.</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

Defines the properties of a HoverVehicle.
**Member Data Documentation**

<table>
<thead>
<tr>
<th>Class</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HoverVehicleData</td>
<td>brakingActivationSpeed</td>
<td>Maximum speed below which a braking force is applied.</td>
</tr>
<tr>
<td>HoverVehicleData</td>
<td>brakingForce</td>
<td>Force generated by braking. The vehicle is considered to be braking if it is moving, but the throttle is off, and no left or right thrust is being applied. This force is only applied when the vehicle's velocity is less than brakingActivationSpeed.</td>
</tr>
<tr>
<td>HoverVehicleData</td>
<td>dragForce</td>
<td>Drag force factor that acts opposite to the vehicle velocity. Also used to determine the vehicle's maxThrustSpeed.</td>
</tr>
</tbody>
</table>

**See also:**
- brakingForce
- mainThrustForce

**ParticleEmitterData HoverVehicleData::dustTrailEmitter**

Emitter to generate particles for the vehicle's dust trail. The trail of dust particles is generated only while the vehicle is moving.
**float HoverVehicleData::dustTrailFreqMod**

Number of dust trail particles to generate based on vehicle speed.

The vehicle's speed is divided by this value to determine how many particles to generate each frame. Lower values give a more dense trail, higher values a more sparse trail.

**Point3F HoverVehicleData::dustTrailOffset**

"X Y Z" offset from the vehicle's origin from which to generate dust trail particles.

By default particles are emitted directly beneath the origin of the vehicle model.

**SFXProfile HoverVehicleData::engineSound**

Looping engine sound.

The volume is dynamically adjusted based on the current thrust level.

**float HoverVehicleData::floatingGravMag**

Scale factor applied to the vehicle gravitational force when the vehicle is floating.

**See also:**

   stabLenMin

**float HoverVehicleData::floatingThrustFactor**
Scalar applied to the vehicle's thrust force when the vehicle is floating.

**Note:**

The `floatingThrustFactor` must be between 0.0 and 1.0 (inclusive).

---

**SFXProfile HoverVehicleData::floatSound**

Looping sound played while the vehicle is floating.

**See also:**

`stabMinLen`

---

**ParticleEmitterData HoverVehicleData::forwardJetEmitter**

Emitter to generate particles for forward jet thrust.

Forward jet thrust particles are emitted from model nodes JetNozzle0 and JetNozzle1.

---

**float HoverVehicleData::gyroDrag**

Damping torque that acts against the vehicle's current angular momentum.

---

**SFXProfile HoverVehicleData::jetSound**

Looping sound played when the vehicle is jetting.
**float HoverVehicleData::mainThrustForce**

Force generated by thrusting the vehicle forward.

Also used to determine the `maxThrustSpeed`:

**Example:**

\[
\text{maxThrustSpeed} = (\text{mainThrustForce} + \text{strafeThrustForce})
\]

**float HoverVehicleData::normalForce**

Force generated in the ground normal direction when the vehicle is not floating (within stabilizer length from the ground).

**See also:**

`stabLenMin`

**float HoverVehicleData::pitchForce**

Pitch (rotation about the X-axis) force applied when steering in the y-axis direction.

**float HoverVehicleData::restorativeForce**

Force generated to stabilize the vehicle (return it to neutral pitch/roll) when the vehicle is floating (more than stabilizer length from the ground).

**See also:**

`stabLenMin`

**float HoverVehicleData::reverseThrustForce**
Force generated by thrusting the vehicle backward.

float HoverVehicleData::rollForce

Roll (rotation about the Y-axis) force applied when steering in the x-axis direction.

float HoverVehicleData::stabDampingConstant

Damping spring force acting against changes in the stabalizer length.

See also:

    stabLenMin

float HoverVehicleData::stabLenMax

Length of the base stabilizer when travelling at maximum speed (maxThrustSpeed).

See also:

    stabLenMin
    mainThrustForce

float HoverVehicleData::stabLenMin

Length of the base stabilizer when travelling at minimum speed (0).

Each tick, the vehicle performs 2 raycasts (from the center back and center front of the vehicle) to check for contact with the ground. The base stabilizer length determines the length of that
raycast; if neither raycast hit the ground, the vehicle is floating, stabilizer spring and ground normal forces are not applied.

**See also:**
- `stabSpringConstant`

<table>
<thead>
<tr>
<th><strong>float HoverVehicleData::stabSpringConstant</strong></th>
</tr>
</thead>
</table>

Value used to generate stabilizer spring force. The force generated depends on stabilizer compression, that is how close the vehicle is to the ground proportional to current stabilizer length.

**See also:**
- `stabLenMin`

<table>
<thead>
<tr>
<th><strong>float HoverVehicleData::steeringForce</strong></th>
</tr>
</thead>
</table>

Yaw (rotation about the Z-axis) force applied when steering in the x-axis direction about the vehicle's Z-axis.

<table>
<thead>
<tr>
<th><strong>float HoverVehicleData::strafeThrustForce</strong></th>
</tr>
</thead>
</table>

Force generated by thrusting the vehicle to one side.

Also used to determine the vehicle's maxThrustSpeed.

**See also:**
**mainThrustForce**

**float HoverVehicleData::triggerTrailHeight**

Maximum height above surface to emit dust trail particles.

If the vehicle is less than `triggerTrailHeight` above a static surface with a material that has 'showDust' set to true, the vehicle will emit particles from the `dustTrailEmitter`.

**float HoverVehicleData::turboFactor**

Scale factor applied to the vehicle's thrust force when jetting.

**float HoverVehicleData::vertFactor**

Scalar applied to the vertical portion of the velocity drag acting on the vehicle.

For the horizontal (X and Y) components of velocity drag, a factor of 0.25 is applied when the vehicle is floating, and a factor of 1.0 is applied when the vehicle is not floating. This velocity drag is multiplied by the vehicle's `dragForce`, as defined above, and the result is subtracted from its movement force.

**Note:**

The `vertFactor` must be between 0.0 and 1.0 (inclusive).

Copyright © GarageGames, LLC. All Rights Reserved.
HTTPObject Class Reference
[Networking]

Allows communications between the game and a server using HTTP.
More...

Inheritance diagram for HTTPObject:

[legend]

List of all members.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>void get (string Address, string requirstURI, string query=&quot;&quot;)</strong></td>
<td>Send a GET command to a server to send or retrieve data.</td>
</tr>
<tr>
<td><strong>void post (string Address, string requirstURI, string query, string post)</strong></td>
<td>Send POST command to a server to send or retrieve data.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Allows communications between the game and a server using HTTP.

**HTTPObject** is derived from **TCPObject** and makes use of the same callbacks for dealing with connections and received data. However, the way in which you use **HTTPObject** to connect with a server is different than **TCPObject**. Rather than opening a connection, sending data, waiting to receive data, and then closing the connection, you issue a **get()** or **post()** and handle the response. The connection is automatically created and destroyed for you.

**Example:**

```plaintext
// In this example we'll retrieve the weather in Las Vegas using Google's API. The response is in XML which could be processed and used by the game using SimXMLDocument, but we'll just output the results to the console in this example.

// Define callbacks for our specific HTTPObject name (WeatherFeed) as the namespace.

// Handle an issue with resolving the server's name
function WeatherFeed::onDNSFailed(%this)
{
    // Store this state
    %this.lastState = "DNSFailed";

    // Handle DNS failure
}

function WeatherFeed::onConnectFailed(%this)
{
    // Store this state
    %this.lastState = "ConnectFailed";
```
function WeatherFeed::onDNSResolved(%this)
{
   // Store this state
   %this.lastState = "DNSResolved";
}

function WeatherFeed::onConnected(%this)
{
   // Store this state
   %this.lastState = "Connected";

   // Clear our buffer
   %this.buffer = "";
}

function WeatherFeed::onDisconnect(%this)
{
   // Store this state
   %this.lastState = "Disconnected";

   // Output the buffer to the console
   echo("Google Weather Results:");
   echo(%this.buffer);
}

// Handle a line from the server
function WeatherFeed::onLine(%this, %line)
{
// Store this line in out buffer
%this.buffer = %this.buffer @ %line;

} // Create the HTTPObject
%feed = new HTTPObject(WeatherFeed);

// Define a dynamic field to store the last connection state
%feed.lastState = "None";

// Send the GET command
%feed.get("www.google.com:80", "/ig/api",

See also:
TCPObject
Member Function Documentation

void HTTPObject::get(string Address, 
  string requirstURI, 
  string query = ""
)

Send a GET command to a server to send or retrieve data.

Parameters:

- **Address**: HTTP web address to send this get call to. Be sure to include the port at the end (IE: "www.garagegames.com:80").
- **requirstURI**: Specific location on the server to access (IE: "index.php").
- **query**: Optional. Actual data to transmit to the server. Can be anything required providing it sticks with limitations of the HTTP protocol. If you were building the URL manually, this is the text that follows the question mark. For example: http://www.google.com/ig/api?weather=Las-Vegas,US

Example:

```cpp
// Create an HTTP object for communication
%httpObj = new HTTPObject();

// Specify a URL to transmit to
%url = "www.garagegames.com:80";

// Specify a URI to communicate with
%URI = "/index.php";

// Specify a query to send.
```
%query = "";

// Send the GET command to the server
%httpObj.get(%url,%URI,%query);

void HTTPObject::post(string Address,
            string requirstURI,
            string query,
            string post
    )

Send POST command to a server to send or retrieve data.

Parameters:

- **Address**: HTTP web address to send this get call to. Be sure to include the port at the end (IE: "www.garagegames.com:80").
- **requirstURI**: Specific location on the server to access (IE: "index.php").
- **query**: Actual data to transmit to the server. Can be anything required providing it sticks with limitations of the HTTP protocol.
- **post**: Submission data to be processed.

Note:
The post() method is currently non-functional.

Example:

```c++
// Create an HTTP object for communication
%httpObj = new HTTPObject();

// Specify a URL to transmit to
%url = "www.garagegames.com:80";
```
// Specify a URI to communicate with
%URI = "/index.php";

// Specify a query to send.
%query = "";

// Specify the submission data.
%post = "";

// Send the POST command to the server
%httpObj.POST(%url,%URI,%query,%post);
InteriorInstance Class Reference
[Game Objects]

Object used to represent buildings and other architectural structures (legacy). More...

Inheritance diagram for InteriorInstance:

```
SimObject

NetObject

SceneObject

InteriorInstance
```

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>void</th>
<th>changeMaterial (string mapTo, Material oldMat, Material newMat)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Change one of the materials on the shape.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>exportToCollada (bool bakeTransform)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Exports the Interior to a Collada file.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>getModelFile ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Get the interior file name.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int</th>
<th>getNumDetailLevels ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Get the number of detail levels interior was created with.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int</th>
<th>getCount (int detailLevel)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Get the number of materials used by interior.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>getName (int detailLevel, int targetNum)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Get the name of the indexed shape material.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>setAlarmMode (string alarmMode)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This sets the alarm mode of the interior.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>setDetailLevel (int level)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Manually changes the current detail level, rather than</td>
</tr>
<tr>
<td></td>
<td>automatically via view distance.</td>
</tr>
</tbody>
</table>
Public Attributes

Media

<table>
<thead>
<tr>
<th>filename</th>
<th>interiorFile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Path and filename of the Interior file (.DIF) to load for this InteriorInstance.</td>
</tr>
</tbody>
</table>

Zoning

<table>
<thead>
<tr>
<th>int</th>
<th>zoneGroup</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID of group the zone is part of.</td>
<td></td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Object used to represent buildings and other architectural structures (legacy).

Interiors are made up entirely from convex hulls or, as they are more commonly known as by game artists, brushes. So what you see is what you collide against. There is no difference between the visible meshes and the collision meshes.

Unlike a DTS or COLLADA mesh, interiors do not support any animation. They also do not support transparent textures. If you need animation or transparency then you are forced to use other model objects.

It is important to note that interiors are no longer the preferred format for large structures. It is an old format, which does not have much to offer above DTS or COLLADA. They are still included in Torque 3D for the sake of backwards compatibility for developers porting older TGE or TGEA projects. It will be deprecated soon.
Member Function Documentation

```cpp
void InteriorInstance::changeMaterial(string mapTo, Material oldMat, Material newMat)
```

Change one of the materials on the shape.

This method changes materials per mapTo with others. The material that is being replaced is mapped to unmapped_mat as a part of this transition.

**Note:**

Warning, right now this only sort of works. It doesn't do a live update like it should.

**Parameters:**

- `mapTo` The name of the material target to remap (from `getTargetName`)
- `oldMat` The old Material that was mapped
- `newMat` The new Material to map

**Example:**

```cpp
// remap the first material in the shape
%mapTo = %interiorObject.getTargetName( 0
%interiorObject.changeMaterial( %mapTo, 0,
```

```cpp
void InteriorInstance::exportToCollada(bool bakeTransform )
```

Exports the Interior to a Collada file.

**Parameters:**

- `bakeTransform` Bakes the InteriorInstance's transform into
the vertex positions

Example:

```
// Export to COLLADA, do not bakeTransform
%interiorObject.exportToCollada(0);
```

string InteriorInstance::getModelFile()

Get the interior file name.

**Returns:**

The name of the interior's model file in .DIF.

Example:

```
%interiorObject.getModelFile();
```

int InteriorInstance::getNumDetailLevels()

Get the number of detail levels interior was created with.

Example:

```
%numLODs = %interiorObject.getNumDetailLevels();
```

int InteriorInstance::getTargetCount(int detailLevel)

Get the number of materials used by interior.

**Parameters:**

*detailLevel* Interior level of detail to scan
Returns:
The number of materials used by the interior at a specified detail level

Example:
```c++
// Find materials used at first level of detail
%targetCount = %interiorObject.getTargetCount(1);
echo(%targetCount);
```

```c++
string InteriorInstance::getTargetName(int detailLevel, int targetNum)
```
Get the name of the indexed shape material.

Parameters:
- `detailLevel` Target LOD
- `targetNum` Index mapped to the target

Returns:
The name of the target (material) at the specified detail level and index

Example:
```c++
// First level of detail, top of the index map
%targetName = %interiorObject.getTargetName(1, 0);
echo(%targetName);
```

```c++
void InteriorInstance::setAlarmMode(string alarmMode )
```
This sets the alarm mode of the interior.

The alarm mode is used when debugging bad geometry for an
interior. When on, the the bad verties will be rendered a different color.

**Parameters:**

*alarmMode* If true the interior will be in an alarm state next frame. Options are 'On' or 'Off'.

**Example:**

```cpp
// Turn on alarm mode debugging for interior
interiorObject.setAlarmMode("On");
```

```cpp
void InteriorInstance::setDetailLevel(int level)
```

Manually changes the current detail level, rather than automatically via view distance.

**Parameters:**

*level* Detail level to force.

**Example:**

```cpp
interiorObject.setDetailLevel(2);
```
Member Data Documentation

<table>
<thead>
<tr>
<th>filename InteriorInstance::interiorFile</th>
</tr>
</thead>
</table>

Path and filename of the Interior file (.DIF) to load for this InteriorInstance.

<table>
<thead>
<tr>
<th>int InteriorInstance::zoneGroup</th>
</tr>
</thead>
</table>

ID of group the zone is part of.

Copyright © GarageGames, LLC. All Rights Reserved.
Item Class Reference
[Game Objects]

Base item class. Uses the ItemData datablock for common properties. More...

Inheritance diagram for Item:

List of all members.
Public Member Functions

string getLastStickyNormal ()
Get the normal of the surface on which the object is stuck.

string getLastStickyPos ()
Get the position on the surface on which this Item is stuck.

bool isAtRest ()
Is the object at rest (ie, no longer moving)?

bool isRotating ()
Is the object still rotating?

bool isStatic ()
Is the object static (ie, non-movable)?

bool setCollisionTimeout (int ignoreColObj=NULL)
Temporarily disable collisions against a specific ShapeBase object.

Callbacks

void onEnterLiquid (string objID, string waterCoverage, string liquidType)
Informs an Item object that it has entered liquid, along with information about the liquid type.

void onLeaveLiquid (string objID, string liquidType)
Informs an Item object that it has left a liquid, along with information about the liquid type.

void onStickyCollision (string objID)
Informs the Item object that it is now sticking to another object.
### Public Attributes

#### Misc

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>rotate</code></td>
<td>If true, the object will automatically rotate around its Z axis.</td>
</tr>
<tr>
<td>bool</td>
<td><code>static</code></td>
<td>If true, the object is not moving in the world.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Enables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Enables selection of all instances of this type.</td>
</tr>
<tr>
<td>static int</td>
<td>maxWarpTicks</td>
<td>When a warp needs to occur due to the client being too far off from the server, this is the maximum number of ticks we'll allow the client to warp to catch up.</td>
</tr>
<tr>
<td>static float</td>
<td>minWarpTicks</td>
<td>Fraction of tick at which instant warp occurs on the client.</td>
</tr>
</tbody>
</table>
Detailed Description

Base Item class. Uses the ItemData datablock for common properties.

Items represent an object in the world, usually one that the player will interact with. One example is a health kit on the group that is automatically picked up when the player comes into contact with it.

Example:

```plaintext
// This is the "health patch" dropped by a datablock ItemData(HealthKitPatch)
{
    // Mission editor category, this datablock will show up in the specified category under the "shapes" category = "Health";
    className = "HealthPatch";

    // Basic Item properties
    shapeFile = "art/shapes/items/patch/healthpatch.dts"
    mass = 2;
    friction = 1;
    elasticity = 0.3;
    emap = true;

    // Dynamic properties used by the script
    pickupName = "a health patch";
    repairAmount = 50;
};

%obj = new Item()
{
    dataBlock = HealthKitSmall;
```
parentGroup = EWCreatorWindow.objectGroup;
static = true;
rotate = true;
};

See also:

ItemData
**Member Function Documentation**

**string Item::getLastStickyNormal( )**

Get the normal of the surface on which the object is stuck.

**Returns:**
Returns The XYZ normal from where this Item is stuck.

**Example:**
```
// Acquire the position where this Item is stuck
%stuckPosition = %item.getLastStickyNormal();
```

**Note:**
Server side only.

**string Item::getLastStickyPos( )**

Get the position on the surface on which this Item is stuck.

**Returns:**
Returns The XYZ position of where this Item is stuck.

**Example:**
```
// Acquire the position where this Item is stuck
%stuckPosition = %item.getLastStickyPos();
```

**Note:**
Server side only.

**bool Item::isAtRest( )**
Is the object at rest (ie, no longer moving)?

**Returns:**
True if the object is at rest, false if it is not.

**Example:**
```c++
// Query the item on if it is or is not at rest.
%isAtRest = %item.isAtRest();
```

```c++
bool Item::isRotating()
```

Is the object still rotating?

**Returns:**
True if the object is still rotating, false if it is not.

**Example:**
```c++
// Query the item on if it is or is not rotating.
%isRotating = %itemData.isRotating();
```

**See also:**
rotate

```c++
bool Item::isStatic()
```

Is the object static (ie, non-movable)?

**Returns:**
True if the object is static, false if it is not.

**Example:**
```c++
// Query the item on if it is or is not static.
```
%isStatic = %itemData.isStatic();

See also:
  static

void Item::onEnterLiquid(string objID,
                      string waterCoverage,
                      string liquidType
              )

Informs an Item object that it has entered liquid, along with information about the liquid type.

Parameters:

  objID          Object ID for this Item object.
  waterCoverage How much coverage of water this Item object has.
  liquidType    The type of liquid that this Item object has entered.

Note:
  Server side only.

See also:
  Item, ItemData, WaterObject

void Item::onLeaveLiquid(string objID,
                      string liquidType
              )

Informs an Item object that it has left a liquid, along with information about the liquid type.

Parameters:
**objID** Object ID for this Item object.

**liquidType** The type of liquid that this Item object has left.

**Note:**
Server side only.

**See also:**
Item, ItemData, WaterObject

```cpp
void Item::onStickyCollision (string objID )
```

Informs the Item object that it is now sticking to another object.

This callback is only called if the ItemData::sticky property for this Item is true.

**Parameters:**
- **objID** Object ID this Item object.

**Note:**
Server side only.

**See also:**
Item, ItemData

```cpp
bool Item::setCollisionTimeout (int ignoreColObj = NULL )
```

Temporarily disable collisions against a specific ShapeBase object.

This is useful to prevent a player from immediately picking up an Item they have just thrown. Only one object may be on the timeout list at a time. The timeout is defined as 15 ticks.

**Parameters:**
objectId ShapeBase object ID to disable collisions against.

Returns:
Returns true if the ShapeBase object requested could be found, false if it could not.

Example:

```
// Set the ShapeBase Object ID to disable
%ignoreColObj = %player.getID();

// Inform this Item object to ignore coll:
%item.setCollisionTimeout(%ignoreColObj);
```
## Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>int Item::maxWarpTicks [static]</strong></td>
<td>When a warp needs to occur due to the client being too far off from the server, this is the maximum number of ticks we'll allow the client to warp to catch up.</td>
</tr>
<tr>
<td><strong>float Item::minWarpTicks [static]</strong></td>
<td>Fraction of tick at which instant warp occurs on the client.</td>
</tr>
<tr>
<td><strong>bool Item::rotate</strong></td>
<td>If true, the object will automatically rotate around its Z axis.</td>
</tr>
<tr>
<td><strong>bool Item::static</strong></td>
<td>If true, the object is not moving in the world.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
**ItemData Class Reference**

[Game Objects]

Stores properties for an individual **Item** type. More...

Inheritance diagram for ItemData:

```
           SimObject
            ↓     ↓
     SimDataBlock          GameBaseData
                        ↓               ↓
     ShapeBaseData        ItemData
                        ↓               ↓
                     ProximityMineData  TurretShapeData
                        ↓               ↓
                     AI-TurretShapeData
```

[legend]

List of all members.
### Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>elasticity</code></td>
<td>A floating-point value specifying how 'bouncy' this ItemData is.</td>
</tr>
<tr>
<td>float</td>
<td><code>friction</code></td>
<td>A floating-point value specifying how much velocity is lost to impact and sliding friction.</td>
</tr>
<tr>
<td>float</td>
<td><code>gravityMod</code></td>
<td>Floating point value to multiply the existing gravity with, just for this ItemData.</td>
</tr>
<tr>
<td>ColorF</td>
<td><code>lightColor</code></td>
<td>Color value to make this light. Example: &quot;1.0,1.0,1.0&quot;.</td>
</tr>
<tr>
<td>bool</td>
<td><code>lightOnlyStatic</code></td>
<td>If true, this ItemData will only cast a light if the Item for this ItemData has a static value of true.</td>
</tr>
<tr>
<td>float</td>
<td><code>lightRadius</code></td>
<td>Distance from the center point of this ItemData for the light to affect.</td>
</tr>
<tr>
<td>int</td>
<td><code>lightTime</code></td>
<td>Time value for the light of this ItemData, used to control the pulse speed of the PulsingLight LightType.</td>
</tr>
<tr>
<td>ItemLightType</td>
<td><code>lightType</code></td>
<td>Type of light to apply to this ItemData. Options are NoLight, ConstantLight, PulsingLight. Default is NoLight.</td>
</tr>
<tr>
<td>float</td>
<td><code>maxVelocity</code></td>
<td>Maximum velocity that this ItemData is able to move.</td>
</tr>
<tr>
<td>bool</td>
<td><code>simpleServerCollision</code></td>
<td></td>
</tr>
</tbody>
</table>
Determines if only simple server-side collision will be used (for pick ups).

<table>
<thead>
<tr>
<th>bool</th>
<th>sticky</th>
</tr>
</thead>
<tbody>
<tr>
<td>If true, ItemData will 'stick' to any surface it collides with.</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

Stores properties for an individual Item type.

Items represent an object in the world, usually one that the player will interact with. One example is a health kit on the group that is automatically picked up when the player comes into contact with it.

ItemData provides the common properties for a set of Items. These properties include a DTS or DAE model used to render the Item in the world, its physical properties for when the Item interacts with the world (such as being tossed by the player), and any lights that emit from the Item.

Example:

datablock ItemData(HealthKitSmall)
{
    category = "Health";
    className = "HealthPatch";
    shapeFile = "art/shapes/items/kit/healthkit.dts";
    gravityMod = "1.0";
    mass = 2;
    friction = 1;
    elasticity = 0.3;
    density = 2;
    drag = 0.5;
    maxVelocity = "10.0";
    emap = true;
    sticky = false;
    dynamicType = "0";
    lightOnlyStatic = false;
    lightType = "NoLight";
    lightColor = "1.0 1.0 1.0 1.0";
    lightTime = 1000;
    lightRadius = 10.0;
simpleServerCollision = true; // Dynamic properties used by the scripts

pickupName = "a small health kit";
repairAmount = 50;
};
Member Data Documentation

**float ItemData::elasticity**

A floating-point value specifying how 'bouncy' this ItemData is.

**float ItemData::friction**

A floating-point value specifying how much velocity is lost to impact and sliding friction.

**float ItemData::gravityMod**

Floating point value to multiply the existing gravity with, just for this ItemData.

**ColorF ItemData::lightColor**

Color value to make this light. Example: "1.0,1.0,1.0".

**See also:**
- lightType

**bool ItemData::lightOnlyStatic**

If true, this ItemData will only cast a light if the Item for this ItemData has a static value of true.

**See also:**
- lightType
### float ItemData::lightRadius

Distance from the center point of this ItemData for the light to affect.

**See also:**

- `lightType`

### int ItemData::lightTime

Time value for the light of this ItemData, used to control the pulse speed of the PulsingLight LightType.

**See also:**

- `lightType`

### ItemLightType ItemData::lightType

Type of light to apply to this ItemData. Options are NoLight, ConstantLight, PulsingLight. Default is NoLight.

### float ItemData::maxVelocity

Maximum velocity that this ItemData is able to move.

### bool ItemData::simpleServerCollision

Determines if only simple server-side collision will be used (for pick ups).

If set to true then only simple, server-side collision detection will be used. This is often the case if the item is used for a pick up object,
such as ammo. If set to false then a full collision volume will be used as defined by the shape. The default is true.

**Note:**

Only applies when using a physics library.

**See also:**

*TurretShape* and *ProximityMine* for examples that should set this to false to allow them to be shot by projectiles.

```cpp
bool ItemData::sticky
```

If true, *ItemData* will 'stick' to any surface it collides with.

When an item does stick to a surface, the *Item::onStickyCollision()* callback is called. The *Item* has methods to retrieve the world position and normal the *Item* is stuck to.

**Note:**

Valid objects to stick to must be of *StaticShapeObjectType*.
LangTable Class Reference
[Localization]

Provides the code necessary to handle the low level management of the string tables for localization. More...

Inheritance diagram for LangTable:

```
+---------------------+     +---------------------+
| SimObject           |     | LangTable            |
|                     |     |                      |
```

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>int addLanguage(string filename,[string languageName])</code></td>
<td>Adds a language to the table.</td>
</tr>
<tr>
<td><code>int getCurrentLanguage()</code></td>
<td>Get the ID of the current language table.</td>
</tr>
<tr>
<td><code>string getLangName(int language)</code></td>
<td>Return the readable name of the language table.</td>
</tr>
<tr>
<td><code>int getNumLang()</code></td>
<td>Used to find out how many languages are in the table.</td>
</tr>
<tr>
<td><code>string getString(string filename)</code></td>
<td>Grabs a string from the specified table.</td>
</tr>
<tr>
<td><code>void setCurrentLanguage(int language)</code></td>
<td>Sets the current language table for grabbing text.</td>
</tr>
<tr>
<td><code>void setDefaultLanguage(int language)</code></td>
<td>Sets the default language table.</td>
</tr>
</tbody>
</table>
Detailed Description

Provides the code necessary to handle the low level management of the string tables for localization.

One LangTable is created for each mod, as well as one for the C++ code. LangTable is responsible for obtaining the correct strings from each and relaying it to the appropriate controls.

See also:

Localization for a full description
Member Function Documentation

**int LangTable::addLanguage (string filename )**

Adds a language to the table.

**Parameters:**
- **filename** Name and path to the language file
- **languageName** Optional name to assign to the new language entry

**Returns:**
True if file was successfully found and language created

**int LangTable::getCurrentLanguage ( )**

Get the ID of the current language table.

**Returns:**
Numerical ID of the current language table

**string LangTable::getLangName (int language )**

Return the readable name of the language table.

**Parameters:**
- **language** Numerical ID of the language table to access

**Returns:**
String containing the name of the table, NULL if ID was invalid or name was never specified
int LangTable::getNumLang()

Used to find out how many languages are in the table.

**Returns:**
Size of the vector containing the languages, numerical

string LangTable::getString(string filename)

Grabs a string from the specified table.
If an invalid is passed, the function will attempt to grab from the default table.

**Parameters:**
`filename` Name of the language table to access

**Returns:**
Text from the specified language table, "" if ID was invalid and default table is not set

void LangTable::setCurrentLanguage(int language)

Sets the current language table for grabbing text.

**Parameters:**
`language` ID of the table

void LangTable::setDefaultLanguage(int language)

Sets the default language table.

**Parameters:**
language ID of the table

Copyright © GarageGames, LLC. All Rights Reserved.
LevelInfo Class Reference
[Miscellaneous]

Stores and controls the rendering and status information for a game level. More...

Inheritance diagram for LevelInfo:

List of all members.
Public Attributes

Lighting

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>advancedLightmapSupport</td>
<td>Enable expanded support for mixing static and dynamic lighting (more costly).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EaseF</td>
<td>ambientLightBlendCurve</td>
<td>Interpolation curve to use for blending from one ambient light color to a different one.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>ambientLightBlendPhase</td>
<td>Number of seconds it takes to blend from one ambient light color to a different one.</td>
</tr>
</tbody>
</table>

LevelInfo

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorI</td>
<td>canvasClearColor</td>
<td>The color used to clear the background before the scene or any GUIs are rendered.</td>
</tr>
</tbody>
</table>

Visibility

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>decalBias</td>
<td>NearPlane bias used when rendering Decal and DecalRoad. This should be tuned to the visibleDistance in your level.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>nearClip</td>
<td>Closest distance from the camera's position to render the world.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>visibleDistance</td>
<td>Furthest distance from the camera's position to render the world.</td>
</tr>
</tbody>
</table>

Fog
<table>
<thead>
<tr>
<th>Variable Type</th>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>fogAtmosphereHeight</td>
<td>A height in meters for altitude fog falloff.</td>
</tr>
<tr>
<td>ColorF</td>
<td>fogColor</td>
<td>The default color for the scene fog.</td>
</tr>
<tr>
<td>float</td>
<td>fogDensity</td>
<td>The 0 to 1 density value for the exponential fog falloff.</td>
</tr>
<tr>
<td>float</td>
<td>fogDensityOffset</td>
<td>An offset from the camera in meters for moving the start of the fog effect.</td>
</tr>
</tbody>
</table>

**Sound**

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SFXAmbience</td>
<td>soundAmbience</td>
<td>The global ambient sound environment.</td>
</tr>
<tr>
<td>SFXDistanceModel</td>
<td>soundDistanceModel</td>
<td>The distance attenuation model to use.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Stores and controls the rendering and status information for a game level.

**Example:**

```java
new LevelInfo(theLevelInfo)
{
    visibleDistance = "1000";
    fogColor = "0.6 0.6 0.7 1";
    fogDensity = "0";
    fogDensityOffset = "700";
    fogAtmosphereHeight = "0";
    canvasClearColor = "0 0 0 255";
    canSaveDynamicFields = "1";
    levelName = "Blank Room";
    desc0 = "A blank room ready to be populated."
    Enabled = "1";
};
```
**Member Data Documentation**

**bool LevelInfo::advancedLightmapSupport**

Enable expanded support for mixing static and dynamic lighting (more costly).

**EaseF LevelInfo::ambientLightBlendCurve**

Interpolation curve to use for blending from one ambient light color to a different one.

**float LevelInfo::ambientLightBlendPhase**

Number of seconds it takes to blend from one ambient light color to a different one.

**ColorI LevelInfo::canvasClearColor**

The color used to clear the background before the scene or any GUIs are rendered.

**float LevelInfo::decalBias**

NearPlane bias used when rendering Decal and DecalRoad. This should be tuned to the visibleDistance in your level.

**float LevelInfo::fogAtmosphereHeight**

A height in meters for altitude fog falloff.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorF LevelInfo::fogColor</td>
<td>The default color for the scene fog.</td>
</tr>
<tr>
<td>float LevelInfo::fogDensity</td>
<td>The 0 to 1 density value for the exponential fog falloff.</td>
</tr>
<tr>
<td>float LevelInfo::fogDensityOffset</td>
<td>An offset from the camera in meters for moving the start of the fog effect.</td>
</tr>
<tr>
<td>float LevelInfo::nearClip</td>
<td>Closest distance from the camera's position to render the world.</td>
</tr>
<tr>
<td>SFXAmbience LevelInfo::soundAmbience</td>
<td>The global ambient sound environment.</td>
</tr>
<tr>
<td>SFXDistanceModel LevelInfo::soundDistanceModel</td>
<td>The distance attenuation model to use.</td>
</tr>
<tr>
<td>float LevelInfo::visibleDistance</td>
<td>Furthest distance from the camera's position to render the world.</td>
</tr>
</tbody>
</table>
LightAnimData Class Reference
[Special Effects, Lighting]

A datablock which defines and performs light animation, such as rotation, brightness fade, and colorization. More...

Inheritance diagram for LightAnimData:

```
    +------------------+
   /                   /
  /                   /
SimObject            SimDataBlock
  |                   |
  |                   |
  V                   V
    +------------------+
```

List of all members.
## Public Attributes

### Brightness

*The brightness animation state.*

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>brightnessA</td>
<td>The value of the A key in the keyframe sequence.</td>
</tr>
<tr>
<td>string</td>
<td>brightnessKeys</td>
<td>The keyframe sequence encoded into a string where characters from A to Z define a position between the two animation values.</td>
</tr>
<tr>
<td>float</td>
<td>brightnessPeriod</td>
<td>The animation time for keyframe sequence.</td>
</tr>
<tr>
<td>bool</td>
<td>brightnessSmooth</td>
<td>If true the transition between keyframes will be smooth.</td>
</tr>
<tr>
<td>float</td>
<td>brightnessZ</td>
<td>The value of the Z key in the keyframe sequence.</td>
</tr>
</tbody>
</table>

### Color

*The RGB color animation state.*

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>colorKeys [3]</td>
<td>The keyframe sequence encoded into a string where characters from A to Z define a position between the two animation values.</td>
</tr>
<tr>
<td>bool</td>
<td>colorSmooth [3]</td>
<td>If true the transition between keyframes will be smooth.</td>
</tr>
</tbody>
</table>
**Offset**

The *XYZ* translation animation state relative to the light position.

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td><code>offsetKeys</code> [3]</td>
<td>The keyframe sequence encoded into a string where characters from A to Z define a position between the two animation values.</td>
</tr>
<tr>
<td>bool</td>
<td><code>offsetSmooth</code> [3]</td>
<td>If true the transition between keyframes will be smooth.</td>
</tr>
</tbody>
</table>

**Rotation**

The *XYZ* rotation animation state relative to the light orientation.

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td><code>rotKeys</code> [3]</td>
<td>The keyframe sequence encoded into a string where characters from A to Z define a position between the two animation values.</td>
</tr>
<tr>
<td>bool</td>
<td><code>rotSmooth</code> [3]</td>
<td>If true the transition between keyframes will be smooth.</td>
</tr>
</tbody>
</table>
float \text{rotZ} [3]

The value of the Z key in the keyframe sequence.
Detailed Description

A datablock which defines and performs light animation, such as rotation, brightness fade, and colorization.

Example:

datablock LightAnimData( SubtlePulseLightAnim )
{
    brightnessA = 0.5;
    brightnessZ = 1;
    brightnessPeriod = 1;
    brightnessKeys = "aza";
    brightnessSmooth = true;
};

See also:

LightBase
LightDescription
**Member Data Documentation**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| float  | `LightAnimData::brightnessA`  
The value of the A key in the keyframe sequence. |
| string | `LightAnimData::brightnessKeys`  
The keyframe sequence encoded into a string where characters from A to Z define a position between the two animation values. |
| float  | `LightAnimData::brightnessPeriod`  
The animation time for keyframe sequence. |
| bool   | `LightAnimData::brightnessSmooth`  
If true the transition between keyframes will be smooth. |
| float  | `LightAnimData::brightnessZ`  
The value of the Z key in the keyframe sequence. |
| float  | `LightAnimData::colorA[3]`  
The value of the A key in the keyframe sequence. |
| string | `LightAnimData::colorKeys[3]`  |
The keyframe sequence encoded into a string where characters from A to Z define a position between the two animation values.

float LightAnimData::colorPeriod[3]

The animation time for keyframe sequence.

bool LightAnimData::colorSmooth[3]

If true the transition between keyframes will be smooth.

float LightAnimData::colorZ[3]

The value of the Z key in the keyframe sequence.

float LightAnimData::offsetA[3]

The value of the A key in the keyframe sequence.

string LightAnimData::offsetKeys[3]

The keyframe sequence encoded into a string where characters from A to Z define a position between the two animation values.

float LightAnimData::offsetPeriod[3]

The animation time for keyframe sequence.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bool LightAnimData::offsetSmooth[3]</code></td>
<td>If true the transition between keyframes will be smooth.</td>
</tr>
<tr>
<td><code>string LightAnimData::rotKeys[3]</code></td>
<td>The keyframe sequence encoded into a string where characters from A to Z define a position between the two animation values.</td>
</tr>
<tr>
<td><code>bool LightAnimData::rotSmooth[3]</code></td>
<td>If true the transition between keyframes will be smooth.</td>
</tr>
</tbody>
</table>
LightBase Class Reference
[Lighting]

This is the base class for light objects. More...

Inheritance diagram for LightBase:

```
  SimObject
   |
   v
  NetObject
   |
   v
  SceneObject
   |
   v
LightBase
   |
   v
  PointLight  SpotLight
```

[legend]

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>void</strong> playAnimation ()</td>
<td>Plays the light animation assigned to this light with the existing <a href="#">LightAnimData</a> datablock.</td>
</tr>
<tr>
<td><strong>void</strong> playAnimation (LightAnimData *anim)</td>
<td>Plays the light animation on this light using a new <a href="#">LightAnimData</a>. If no <a href="#">LightAnimData</a> is passed the existing one is played.</td>
</tr>
<tr>
<td><strong>void</strong> setLightEnabled (bool state)</td>
<td>Toggles the light on and off.</td>
</tr>
</tbody>
</table>
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td><code>pauseAnimation</code></td>
<td>Stops the light animation.</td>
</tr>
</tbody>
</table>

## Light Animation

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>animate</code></td>
<td>Toggles animation for the light on and off.</td>
</tr>
<tr>
<td>float</td>
<td><code>animationPeriod</code></td>
<td>The length of time in seconds for a single playback of the light animation (must be &gt; 0).</td>
</tr>
<tr>
<td>float</td>
<td><code>animationPhase</code></td>
<td>The phase used to offset the animation start time to vary the animation of nearby lights.</td>
</tr>
</tbody>
</table>

## LightAnimData

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datablock</td>
<td><code>animationType</code></td>
<td>Datablock containing light animation information (LightAnimData).</td>
</tr>
</tbody>
</table>

## Advanced Lighting

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point3F</td>
<td><code>attenuationRatio</code></td>
<td>The proportions of constant, linear, and quadratic attenuation to use for the falloff for point and spot lights.</td>
</tr>
<tr>
<td>filename</td>
<td><code>cookie</code></td>
<td>A custom pattern texture which is projected from the light.</td>
</tr>
<tr>
<td>float</td>
<td><code>fadeStartDistance</code></td>
<td>Start fading shadows out at this distance. 0 = auto calculate this distance.</td>
</tr>
<tr>
<td>bool</td>
<td><code>lastSplitTerrainOnly</code></td>
<td>This toggles only terrain being rendered to the last</td>
</tr>
</tbody>
</table>
split of a PSSM shadow map.

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>logWeight</td>
<td>The logarithmic PSSM split distance factor.</td>
</tr>
<tr>
<td>int</td>
<td>numSplits</td>
<td>The logarithmic PSSM split distance factor.</td>
</tr>
<tr>
<td>Point4F</td>
<td>overDarkFactor</td>
<td>The ESM shadow darkening factor.</td>
</tr>
<tr>
<td>float</td>
<td>shadowDistance</td>
<td>The distance from the camera to extend the PSSM shadow.</td>
</tr>
<tr>
<td>float</td>
<td>shadowSoftness</td>
<td></td>
</tr>
<tr>
<td>int</td>
<td>texSize</td>
<td>The texture size of the shadow map.</td>
</tr>
</tbody>
</table>

**ShadowType**

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ShadowType</td>
<td>shadowType</td>
<td>The type of shadow to use on this light.</td>
</tr>
</tbody>
</table>

**Light**

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>brightness</td>
<td>Adjusts the lights power, 0 being off completely.</td>
</tr>
<tr>
<td>bool</td>
<td>castShadows</td>
<td>Enables/disabled shadow casts by this light.</td>
</tr>
<tr>
<td>ColorF</td>
<td>color</td>
<td>Changes the base color hue of the light.</td>
</tr>
<tr>
<td>bool</td>
<td>isEnabled</td>
<td>Enables/Disables the object rendering and functionality in the scene.</td>
</tr>
<tr>
<td>float</td>
<td>priority</td>
<td>Used for sorting of lights by the light manager. Priority determines if a light has a stronger effect than, those with a lower value.</td>
</tr>
</tbody>
</table>
**Misc**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>flareScale</td>
<td>Globally scales all features of the light flare.</td>
</tr>
</tbody>
</table>

**LightFlareData**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datablock</td>
<td>flareType</td>
<td>Datablock containing light flare information (LightFlareData).</td>
</tr>
</tbody>
</table>

**Advanced Lighting Lightmap**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>includeLightmappedGeometryInShadow</td>
<td>This light should render lightmapped geometry during its shadow-map update (ignored if 'representedInLightmap' is false).</td>
</tr>
<tr>
<td>bool</td>
<td>representedInLightmap</td>
<td>This light is represented in lightmaps (static light, default: false).</td>
</tr>
<tr>
<td>ColorF</td>
<td>shadowDarkenColor</td>
<td>The color that should be used to multiply-blend dynamic shadows onto lightmapped geometry (ignored if 'representedInLightmap' is false).</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>static bool</th>
<th>isRenderable</th>
<th>Disables rendering of all instances of this type.</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

This is the base class for light objects.

It is *NOT* intended to be used directly in script, but exists to provide the base member variables and generic functionality. You should be using the derived classes PointLight and SpotLight, which can be declared in TorqueScript or added from the World Editor.

For this class, we only add basic lighting options that all lighting systems would use. The specific lighting system options are injected at runtime by the lighting system itself.

See also:

PointLight
SpotLight
Member Function Documentation

void LightBase::playAnimation()

Plays the light animation assigned to this light with the existing LightAnimData datablock.

Example:

// Play the animation assigned to this light
CrystalLight.playAnimation();

void LightBase::playAnimation(LightAnimData anim)

Plays the light animation on this light using a new LightAnimData. If no LightAnimData is passed the existing one is played.

Parameters:

anim  Name of the LightAnimData datablock to be played

Example:

// Play the animation using a new LightAnimData
CrystalLight.playAnimation(SubtlePulseLightAnim);

void LightBase::setLightEnabled(bool state)

Toggles the light on and off.

Parameters:

state  Turns the light on (true) or off (false)

Example:

// Disable the light
CrystalLight.setLightEnabled(false);

// Renable the light
CrystalLight.setLightEnabled(true);
### Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>LightBase::animate</td>
<td>Toggles animation for the light on and off.</td>
</tr>
<tr>
<td>float</td>
<td>LightBase::animationPeriod</td>
<td>The length of time in seconds for a single playback of the light animation.</td>
</tr>
<tr>
<td>float</td>
<td>LightBase::animationPhase</td>
<td>The phase used to offset the animation start time to vary the animation.</td>
</tr>
<tr>
<td>LightAnimData</td>
<td>LightBase::animationType</td>
<td>Datablock containing light animation information (<a href="#">LightAnimData</a>).</td>
</tr>
<tr>
<td>Point3F</td>
<td>LightBase::attenuationRatio</td>
<td>The proportions of constant, linear, and quadratic attenuation to use for the falloff for point and spot lights.</td>
</tr>
<tr>
<td>float</td>
<td>LightBase::brightness</td>
<td>Adjusts the lights power, 0 being off completely.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>bool LightBase::castShadows</strong></td>
<td>Enables/disabled shadow casts by this light.</td>
<td></td>
</tr>
<tr>
<td><strong>ColorF LightBase::color</strong></td>
<td>Changes the base color hue of the light.</td>
<td></td>
</tr>
<tr>
<td><strong>filename LightBase::cookie</strong></td>
<td>A custom pattern texture which is projected from the light.</td>
<td></td>
</tr>
<tr>
<td><strong>float LightBase::fadeStartDistance</strong></td>
<td>Start fading shadows out at this distance. 0 = auto calculate this distance.</td>
<td></td>
</tr>
<tr>
<td><strong>float LightBase::flareScale</strong></td>
<td>Globally scales all features of the light flare.</td>
<td></td>
</tr>
<tr>
<td><strong>LightFlareData LightBase::flareType</strong></td>
<td>Datablock containing light flare information (LightFlareData).</td>
<td></td>
</tr>
<tr>
<td><strong>bool LightBase::includeLightmappedGeometryInShadow</strong></td>
<td>This light should render lightmapped geometry during its shadow-map update (ignored if 'representedInLightmap' is false).</td>
<td></td>
</tr>
</tbody>
</table>
**bool LightBase::isEnabled**

Enables/Disables the object rendering and functionality in the scene.

**bool LightBase::lastSplitTerrainOnly**

This toggles only terrain being rendered to the last split of a PSSM shadow map.

**float LightBase::logWeight**

The logarithmic PSSM split distance factor.

**int LightBase::numSplits**

The logarithmic PSSM split distance factor.

**Point4F LightBase::overDarkFactor**

The ESM shadow darkening factor.

**void LightBase::pauseAnimation**

Stops the light animation.

**float LightBase::priority**

Used for sorting of lights by the light manager. Priority determines
if a light has a stronger effect than, those with a lower value.

```
bool LightBase::representedInLightmap
```

This light is represented in lightmaps (static light, default: false).

```
ColorF LightBase::shadowDarkenColor
```

The color that should be used to multiply-blend dynamic shadows onto lightmapped geometry (ignored if 'representedInLightmap' is false).

```
float LightBase::shadowDistance
```

The distance from the camera to extend the PSSM shadow.

```
float LightBase::shadowSoftness
```

```
ShadowType LightBase::shadowType
```

The type of shadow to use on this light.

```
int LightBase::texSize
```

The texture size of the shadow map.
LightDescription Class Reference
[Lighting]

A helper datablock used by classes (such as shapebase) that submit lights to the scene but do not use actual "LightBase" objects. More...

Inheritance diagram for LightDescription:

```
SimObject

SimDataBlock

LightDescription
```

List of all members.
Public Member Functions

```cpp
void apply()

Force an inspectPostApply call for the benefit of tweaking via the console.
```
## Public Attributes

### Light Animation

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>animationPeriod</td>
<td>The length of time in seconds for a single playback of the light animation.</td>
</tr>
<tr>
<td>float</td>
<td>animationPhase</td>
<td>The phase used to offset the animation start time to vary the animation of nearby lights.</td>
</tr>
</tbody>
</table>

**LightAnimData**

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data block</td>
<td>animationType</td>
<td>Datablock containing light animation information (<a href="#">LightAnimData</a>).</td>
</tr>
</tbody>
</table>

### Advanced Lighting

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point3F</td>
<td>attenuationRatio</td>
<td>The proportions of constant, linear, and quadratic attenuation to use for the falloff for point and spot lights.</td>
</tr>
<tr>
<td>filename</td>
<td>cookie</td>
<td>A custom pattern texture which is projected from the light.</td>
</tr>
<tr>
<td>float</td>
<td>fadeStartDistance</td>
<td>Start fading shadows out at this distance. 0 = auto calculate this distance.</td>
</tr>
<tr>
<td>bool</td>
<td>lastSplitTerrainOnly</td>
<td>This toggles only terrain being rendered to the last split of a PSSM shadow map.</td>
</tr>
<tr>
<td>float</td>
<td>logWeight</td>
<td>The logarithmic PSSM split distance factor.</td>
</tr>
<tr>
<td>int</td>
<td>numSplits</td>
<td></td>
</tr>
</tbody>
</table>
The logarithmic PSSM split distance factor.

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point4F</td>
<td>overDarkFactor</td>
<td>The ESM shadow darkening factor.</td>
</tr>
<tr>
<td>float</td>
<td>shadowDistance</td>
<td>The distance from the camera to extend the PSSM shadow.</td>
</tr>
<tr>
<td>float</td>
<td>shadowSoftness</td>
<td></td>
</tr>
<tr>
<td>ShadowType</td>
<td>shadowType</td>
<td>The type of shadow to use on this light.</td>
</tr>
<tr>
<td>int</td>
<td>texSize</td>
<td>The texture size of the shadow map.</td>
</tr>
</tbody>
</table>

**Light**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>brightness</td>
<td>Adjusts the lights power, 0 being off completely.</td>
</tr>
<tr>
<td>bool</td>
<td>castShadows</td>
<td>Enables/disabled shadow casts by this light.</td>
</tr>
<tr>
<td>ColorF</td>
<td>color</td>
<td>Changes the base color hue of the light.</td>
</tr>
<tr>
<td>float</td>
<td>range</td>
<td>Controls the size (radius) of the light.</td>
</tr>
</tbody>
</table>

**Misc**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>flareScale</td>
<td>Globally scales all features of the light flare.</td>
</tr>
<tr>
<td>LightFlareData</td>
<td>flareType</td>
<td>Datablock containing light flare information (LightFlareData).</td>
</tr>
</tbody>
</table>

**Advanced Lighting Lightmap**
<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>includeLightmappedGeometryInShadow</code></td>
<td>This light should render lightmapped geometry during its shadow-map update (ignored if 'representedInLightmap' is false).</td>
</tr>
<tr>
<td>bool</td>
<td><code>representedInLightmap</code></td>
<td>This light is represented in lightmaps (static light, default: false).</td>
</tr>
<tr>
<td>ColorF</td>
<td><code>shadowDarkenColor</code></td>
<td>The color that should be used to multiply-blend dynamic shadows onto lightmapped geometry (ignored if 'representedInLightmap' is false).</td>
</tr>
</tbody>
</table>
Detailed Description

A helper datablock used by classes (such as shapebase) that submit lights to the scene but do not use actual "LightBase" objects.

This datablock stores the properties of that light as fields that can be initialized from script.

Example:

```plaintext
// Declare a light description to be used datablock LightDescription(RocketLauncherLightDesc)
{
    range = 4.0;
    color = "1 1 0";
    brightness = 5.0;
    animationType = PulseLightAnim;
    animationPeriod = 0.25;
};

// Declare a ProjectileDatablock which uses the light description datablock ProjectileData(RocketLauncherProjectile)
{
    lightDesc = RocketLauncherLightDesc;
    projectileShapeName = "art/shapes/weapons/SwarmGun/rocket.dts"
    directDamage = 30;
    radiusDamage = 30;
    damageRadius = 5;
    areaImpulse = 2500;

    // ... remaining ProjectileData fields
};
```
See also:

LightBase
Member Function Documentation

```cpp
void LightDescription::apply()
```

Force an inspectPostApply call for the benefit of tweaking via the console.

Normally this functionality is only exposed to objects via the World Editor, once changes have been made. Exposing apply to script allows you to make changes to it on the fly without the World Editor.

**Note:**

This is intended for debugging and tweaking, not for game play

**Example:**

```cpp
// Change a property of the light description
RocketLauncherLightDesc.brightness = 10;

// Make it so
RocketLauncherLightDesc.apply();
```
Member Data Documentation

float LightDescription::animationPeriod

The length of time in seconds for a single playback of the light animation.

float LightDescription::animationPhase

The phase used to offset the animation start time to vary the animation of nearby lights.

LightAnimData LightDescription::animationType

Datablock containing light animation information (LightAnimData).

Point3F LightDescription::attenuationRatio

The proportions of constant, linear, and quadratic attenuation to use for the falloff for point and spot lights.

float LightDescription::brightness

Adjusts the lights power, 0 being off completely.

bool LightDescription::castShadows

Enables/disabled shadow casts by this light.
<table>
<thead>
<tr>
<th><strong>ColorF LightDescription::color</strong></th>
<th>Changes the base color hue of the light.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>filename LightDescription::cookie</strong></td>
<td>A custom pattern texture which is projected from the light.</td>
</tr>
<tr>
<td><strong>float LightDescription::fadeStartDistance</strong></td>
<td>Start fading shadows out at this distance. 0 = auto calculate this distance.</td>
</tr>
<tr>
<td><strong>float LightDescription::flareScale</strong></td>
<td>Globally scales all features of the light flare.</td>
</tr>
<tr>
<td><strong>LightFlareData LightDescription::flareType</strong></td>
<td>Datablock containing light flare information (<strong>LightFlareData</strong>).</td>
</tr>
<tr>
<td><strong>bool LightDescription::includeLightmappedGeometryInShadow</strong></td>
<td>This light should render lightmapped geometry during its shadow-map update (ignored if 'representedInLightmap' is false).</td>
</tr>
<tr>
<td><strong>bool LightDescription::lastSplitTerrainOnly</strong></td>
<td>This toggles only terrain being rendered to the last split of a PSSM shadow map.</td>
</tr>
<tr>
<td>Variable Type</td>
<td>Variable Name</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td>float</td>
<td>LightDescription::logWeight</td>
</tr>
<tr>
<td>int</td>
<td>LightDescription::numSplits</td>
</tr>
<tr>
<td>Point4F</td>
<td>LightDescription::overDarkFactor</td>
</tr>
<tr>
<td>float</td>
<td>LightDescription::range</td>
</tr>
<tr>
<td>bool</td>
<td>LightDescription::representedInLightmap</td>
</tr>
<tr>
<td>ColorF</td>
<td>LightDescription::shadowDarkenColor</td>
</tr>
<tr>
<td>float</td>
<td>LightDescription::shadowDistance</td>
</tr>
</tbody>
</table>
The distance from the camera to extend the PSSM shadow.

float LightDescription::shadowSoftness

ShadowType LightDescription::shadowType

The type of shadow to use on this light.

int LightDescription::texSize

The texture size of the shadow map.

Copyright © GarageGames, LLC. All Rights Reserved.
LightFlareData Class Reference

[Lighting]

Defines a light flare effect usable by scene lights. More...

Inheritance diagram for LightFlareData:

List of all members.
Public Member Functions

void apply ()

   Intended as a helper to developers and editor scripts.
Public Attributes

FlareElements

<table>
<thead>
<tr>
<th>data type</th>
<th>variable name</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>elementDist [20]</td>
<td>Where this element appears along the flare beam.</td>
</tr>
<tr>
<td>bool</td>
<td>elementRotate [20]</td>
<td>Defines if this element orients to point along the flare beam or if it is always upright.</td>
</tr>
<tr>
<td>float</td>
<td>elementScale [20]</td>
<td>Size scale applied to this element.</td>
</tr>
<tr>
<td>ColorF</td>
<td>elementTint [20]</td>
<td>Used to modulate this element's color if elementUseLightColor is false.</td>
</tr>
<tr>
<td>bool</td>
<td>elementUseLightColor [20]</td>
<td>If true this element's color is modulated by the light color. If false, elementTint will be used.</td>
</tr>
<tr>
<td>bool</td>
<td>flareEnabled</td>
<td>Allows the user to disable this flare globally for any lights referencing it.</td>
</tr>
<tr>
<td>filename</td>
<td>flareTexture</td>
<td>The texture / sprite sheet for this flare.</td>
</tr>
</tbody>
</table>

LightFlareData

<table>
<thead>
<tr>
<th>data type</th>
<th>variable name</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>occlusionRadius</td>
<td>If positive an occlusion query is used to test flare visibility, else it uses simple raycasts.</td>
</tr>
<tr>
<td>float</td>
<td>overallScale</td>
<td>Size scale applied to all elements of the flare.</td>
</tr>
</tbody>
</table>
**bool** renderReflectPass

If false the flare does not render in reflections, else only non-zero distance elements are rendered.
**Detailed Description**

Defines a light flare effect usable by scene lights.

LightFlareData is a datablock which defines a type of flare effect. This may then be referenced by other classes which support the rendering of a flare: Sun, ScatterSky, LightBase.

A flare contains one or more elements defined in the element* named fields of LightFlareData, with a maximum of ten elements. Each element is rendered as a 2D sprite in screenspace.

**Example:**

```csharp
// example from Full Template, core/art/datablock
.datablock LightFlareData( LightFlareExample )
{
    overallScale = 2.0;
    flareEnabled = true;
    renderReflectPass = true;
    flareTexture = "./../special/lensFlareSheet1"
    occlusionRadius = 0.25;

    elementRect[0] = "0 512 512 512";
    elementDist[0] = 0.0;
    elementScale[0] = 0.5;
    elementTint[0] = "1.0 1.0 1.0";
    elementRotate[0] = false;
    elementUseLightColor[0] = false;

    elementRect[1] = "512 0 512 512";
    elementDist[1] = 0.0;
    elementScale[1] = 2.0;
    elementTint[1] = "0.5 0.5 0.5";
    elementRotate[1] = false;
}
The `elementDist` field defines where along the flare's beam the element appears. A distance of 0.0 is directly over the light source, a distance of 1.0 is at the screen center, and a distance of 2.0 is at the position of the light source mirrored across the screen center.
Member Function Documentation

`void LightFlareData::apply()`

Intended as a helper to developers and editor scripts.

Force trigger an `inspectPostApply`
**Member Data Documentation**

```plaintext
float LightFlareData::elementDist[20]
```

Where this element appears along the flare beam.

```plaintext
RectF LightFlareData::elementRect[20]
```

A rectangle specified in pixels of the flareTexture image.

```plaintext
bool LightFlareData::elementRotate[20]
```

Defines if this element orients to point along the flare beam or if it is always upright.

```plaintext
float LightFlareData::elementScale[20]
```

Size scale applied to this element.

```plaintext
ColorF LightFlareData::elementTint[20]
```

Used to modulate this element's color if elementUseLightColor is false.

**See also:**

  elementUseLightColor

```plaintext
bool LightFlareData::elementUseLightColor[20]
```

If true this element's color is modulated by the light color. If false,
elementTint will be used.

**See also:**

* elementTint

---

**bool LightFlareData::flareEnabled**

Allows the user to disable this flare globally for any lights referencing it.

**filename LightFlareData::flareTexture**

The texture / sprite sheet for this flare.

**float LightFlareData::occlusionRadius**

If positive an occlusion query is used to test flare visibility, else it uses simple raycasts.

**float LightFlareData::overallScale**

Size scale applied to all elements of the flare.

**bool LightFlareData::renderReflectPass**

If false the flare does not render in reflections, else only non-zero distance elements are rendered.
An emitter for lightning bolts. More...

Inheritance diagram for Lightning:

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void <code>strikeObject</code> (int id=NULL)</td>
<td>Creates a LightningStrikeEvent which strikes a specific object.</td>
</tr>
<tr>
<td>void <code>strikeRandomPoint</code> ()</td>
<td>Creates a LightningStrikeEvent which attempts to strike and damage a random object in range of the Lightning object.</td>
</tr>
<tr>
<td>void <code>warningFlashes</code> ()</td>
<td>Creates a LightningStrikeEvent that triggers harmless lightning bolts on all clients. No objects will be damaged by these bolts.</td>
</tr>
</tbody>
</table>

### Callbacks

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void <code>applyDamage</code> (Point3F hitPosition, Point3F hitNormal, SceneObject hitObject)</td>
<td>Informs an object that it was hit by a lightning bolt and needs to take damage.</td>
</tr>
</tbody>
</table>
## Public Attributes

### Bolts

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>boltStartRadius</td>
<td>Radial distance from the center of the Lightning object for the start point of the bolt.</td>
</tr>
<tr>
<td>float</td>
<td>chanceToHitTarget</td>
<td>Percentage chance (0-1) that a given lightning bolt will hit something.</td>
</tr>
<tr>
<td>bool</td>
<td>useFog</td>
<td>Controls whether lightning bolts are affected by fog when they are rendered.</td>
</tr>
</tbody>
</table>

### Colors

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorF</td>
<td>color</td>
<td>Color to blend the strike texture with.</td>
</tr>
<tr>
<td>ColorF</td>
<td>fadeColor</td>
<td>Color to blend the strike texture with when the bolt is fading away.</td>
</tr>
</tbody>
</table>

### Strikes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>strikeRadius</td>
<td>Horizontal size (XY plane) of the search box used to find and damage Player or Vehicle objects within range of the strike.</td>
</tr>
<tr>
<td>int</td>
<td>strikesPerMinute</td>
<td>Number of lightning strikes to perform per minute.</td>
</tr>
<tr>
<td>float</td>
<td>strikeWidth</td>
<td>Width of a lightning bolt.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
**Detailed Description**

An emitter for lightning bolts.

*Lightning* strike events are created on the server and transmitted to all clients to render the bolt. The strike may be followed by a random thunder sound. *Player* or *Vehicle* objects within the *Lightning* strike range can be hit and damaged by bolts.

**See also:**

*LightningData*
Member Function Documentation

```cpp
void Lightning::applyDamage(Point3F hitPosition,
                            Point3F hitNormal,
                            SceneObject hitObject)
```

Informs an object that it was hit by a lightning bolt and needs to take damage.

**Parameters:**

- `hitPosition` World position hit by the lightning bolt.
- `hitNormal` Surface normal at `hitPosition`.
- `hitObject` Player or Vehicle object that was hit.

**Example:**

```cpp
function Lightning::applyDamage( %this, %hitPosition, %hitNormal, %hitObject )
{
    // apply damage to the player
    %hitObject.applyDamage( 25 );
}
```

```cpp
void Lightning::strikeObject(int id = NULL)
```

Creates a LightningStrikeEvent which strikes a specific object.

**Note:**

This method is currently unimplemented.

```cpp
void Lightning::strikeRandomPoint()
```

Creates a LightningStrikeEvent which attempts to strike and
damage a random object in range of the Lightning object.

**Example:**

```// Generate a damaging lightning strike effect
%lightning.strikeRandomPoint();```

```void Lightning::warningFlashes( )

Creates a LightningStrikeEvent that triggers harmless lightning bolts on all clients. No objects will be damaged by these bolts.

**Example:**

```// Generate a harmless lightning strike effect
%lightning.warningFlashes();```
**Member Data Documentation**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>Lightning::boltStartRadius</td>
<td>Radial distance from the center of the Lightning object for the start point of the bolt. The actual start point will be a random point within this radius.</td>
</tr>
<tr>
<td>float</td>
<td>Lightning::chanceToHitTarget</td>
<td>Percentage chance (0-1) that a given lightning bolt will hit something.</td>
</tr>
<tr>
<td>ColorF</td>
<td>Lightning::color</td>
<td>Color to blend the strike texture with.</td>
</tr>
<tr>
<td>ColorF</td>
<td>Lightning::fadeColor</td>
<td>Color to blend the strike texture with when the bolt is fading away.</td>
</tr>
<tr>
<td>float</td>
<td>Lightning::strikeRadius</td>
<td>Horizontal size (XY plane) of the search box used to find and damage Player or Vehicle objects within range of the strike. Only the object at highest altitude with a clear line of sight to the bolt will be hit.</td>
</tr>
<tr>
<td>Type</td>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>int</td>
<td>Lightning::strikesPerMinute</td>
<td>Number of lightning strikes to perform per minute. Automatically invokes strikeRandomPoint() at regular intervals.</td>
</tr>
<tr>
<td>float</td>
<td>Lightning::strikeWidth</td>
<td>Width of a lightning bolt.</td>
</tr>
<tr>
<td>bool</td>
<td>Lightning::useFog</td>
<td>Controls whether lightning bolts are affected by fog when they are rendered.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
LightningData Class Reference
[Special Effects, Atmosphere]

Common data for a Lightning emitter object. More...

Inheritance diagram for LightningData:

```
  +---------------------------+
  | SimObject                |
  |                           |
  +---------------------------+   +---------------------------+
                           |   | SimDataBlock             |
                           +---------------------------+   +---------------------------+
                                    |   | GameBaseData             |
                                    +---------------------------+   +---------------------------+
                                                |   | LightningData            |
                                                +---------------------------+
```

Legend

List of all members.
### Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SFXTrack</td>
<td>strikeSound</td>
<td>Sound profile to play when a lightning strike occurs.</td>
</tr>
<tr>
<td>string</td>
<td>strikeTextures</td>
<td>List of textures to use to render lightning strikes.</td>
</tr>
<tr>
<td>SFXTrack</td>
<td>thunderSounds</td>
<td>List of thunder sound effects to play.</td>
</tr>
</tbody>
</table>
Detailed Description

Common data for a Lightning emitter object.

See also:
  Lightning
## Member Data Documentation

<table>
<thead>
<tr>
<th>SFXTrack LightningData::strikeSound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sound profile to play when a lightning strike occurs.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string LightningData::strikeTextures[8]</th>
</tr>
</thead>
<tbody>
<tr>
<td>List of textures to use to render lightning strikes.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SFXTrack LightningData::thunderSounds[8]</th>
</tr>
</thead>
<tbody>
<tr>
<td>List of thunder sound effects to play.</td>
</tr>
<tr>
<td>A random one of these sounds will be played shortly after each strike occurs.</td>
</tr>
</tbody>
</table>
LightningStrikeEvent Class Reference

[Special Effects, Atmosphere]

Network event that triggers a lightning strike on the client when it is received. More...
**Detailed Description**

Network event that triggers a lightning strike on the client when it is received.

This event is sent to all clients when the `warningFlashes()`, `strikeRandomPoint()` or `strikeObject()` methods are invoked on the `Lightning` object on the server.

**See also:**
- Lightning, LightningData
Marker Class Reference
[Miscellaneous]

A single joint, or knot, along a path. Should be stored inside a Path container object. A path markers can be one of three primary movement types: "normal", "Position Only", or "Kink". More...

Inheritance diagram for Marker:

```
  SimObject
   ↓
  NetObject
   ↓
SceneObject
   ↓
Marker
```

List of all members.
### Public Attributes

#### Misc

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>msToNext</td>
<td>Milliseconds to next marker in sequence.</td>
</tr>
<tr>
<td>int</td>
<td>seqNum</td>
<td>Marker position in sequence of markers on this path.</td>
</tr>
<tr>
<td>MarkerSmoothingType</td>
<td>smoothingType</td>
<td>Path smoothing at this marker/knot. &quot;Linear&quot; means no smoothing, while &quot;Spline&quot; means to smooth.</td>
</tr>
<tr>
<td>MarkerKnotType</td>
<td>type</td>
<td>Type of this marker/knot. A &quot;normal&quot; knot will have a smooth camera translation/rotation effect.</td>
</tr>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>--------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Disables rendering of all instances of this type.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static bool</th>
<th>isSelectable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

A single joint, or knot, along a path. Should be stored inside a Path container object. A path markers can be one of three primary movement types: "normal", "Position Only", or "Kink".

Example:

```java
new path()
{
    isLooping = "1";

    new Marker()
    {
        seqNum = "0";
        type = "Normal";
        msToNext = "1000";
        smoothingType = "Spline";
        position = "-0.054708 -35.0612 234.802"
        rotation = "1 0 0 0"
    }
};
```

See also:

Path
# Member Data Documentation

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int Marker::msToNext</td>
<td>Milliseconds to next marker in sequence.</td>
</tr>
<tr>
<td>int Marker::seqNum</td>
<td>Marker position in sequence of markers on this path.</td>
</tr>
<tr>
<td>MarkerSmoothingType Marker::smoothingType</td>
<td>Path smoothing at this marker/knot. &quot;Linear&quot; means no smoothing, while &quot;Spline&quot; means to smooth.</td>
</tr>
<tr>
<td>MarkerKnotType Marker::type</td>
<td>Type of this marker/knot. A &quot;normal&quot; knot will have a smooth camera translation/rotation effect. &quot;Position Only&quot; will do the same for translations, leaving rotation un-touched. Lastly, a &quot;Kink&quot; means the rotation will take effect immediately for an abrupt rotation change.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
Material Class Reference
[GFX]

A material in Torque 3D is a data structure that describes a surface. More...

Inheritance diagram for Material:

List of all members.
### Public Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int alphaRef</td>
<td>The alpha reference value for alpha testing. Must be between 0 to 255.</td>
</tr>
<tr>
<td>bool alphaTest</td>
<td>Enables alpha test when rendering the material.</td>
</tr>
<tr>
<td>MaterialAnimType</td>
<td><strong>animFlags</strong> [4] The types of animation to play on this material.</td>
</tr>
<tr>
<td>bool bumpAtlas</td>
<td>[4]</td>
</tr>
<tr>
<td>bool castShadows</td>
<td>If set to false the lighting system will not cast shadows from this material.</td>
</tr>
<tr>
<td>Point2I cellIndex</td>
<td>[4]</td>
</tr>
<tr>
<td>Point2I cellLayout</td>
<td>[4]</td>
</tr>
<tr>
<td>int cellSize</td>
<td>[4]</td>
</tr>
<tr>
<td>string cubemap</td>
<td>The name of a <strong>CubemapData</strong> for environment mapping.</td>
</tr>
<tr>
<td>filename detailNormalMap</td>
<td>[4] A second normal map texture applied at the</td>
</tr>
<tr>
<td>Type</td>
<td>Name</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>float</td>
<td>detailNormalMapStrength [4]</td>
</tr>
<tr>
<td>ColorF</td>
<td>diffuseColor [4]</td>
</tr>
<tr>
<td>bool</td>
<td>doubleSided</td>
</tr>
<tr>
<td>void</td>
<td>dumpInstances</td>
</tr>
<tr>
<td>bool</td>
<td>dynamicCubemap</td>
</tr>
</tbody>
</table>
### filename envTex [4]
For backwards compatibility.

### void flush
Flushes all material instances that use this material.

### string getAnimFlags

### string getFilename
Get filename of material.

### bool glow [4]
Enables rendering this material to the glow buffer.

### bool isAutoGenerated
Returns true if this Material was automatically generated by MaterialList::mapMaterials().

### filename lightMap [4]
The lightmap texture used with pureLight.

### string mapTo
Used to map this material to the material name used by TSShape.

### float minnaertConstant [4]
The Minnaert shading constant value. Must be greater than 0 to enable the effect.

### filename normalMap [4]
The normal map texture. You can use the DXTnm format only when per-pixel specular highlights are disabled, or a specular map is in use.

### filename overlayMap [4]
A secondary diffuse color texture map which will use the second texcoord of a mesh.
<table>
<thead>
<tr>
<th><strong>filename</strong></th>
<th><strong>overlayTex</strong> [4]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>For backwards compatibility.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong></th>
<th><strong>parallaxScale</strong> [4]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Enables parallax mapping and defines the scale factor for the parallax effect. Typically this value is less than 0.4 else the effect breaks down.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>bool</strong></th>
<th><strong>pixelSpecular</strong> [4]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This enables per-pixel specular highlights controlled by the alpha channel of the normal map texture. Note that if pixel specular is enabled the DXTnm format will not work with your normal map, unless you are also using a specular map.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>bool</strong></th>
<th><strong>planarReflection</strong></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th><strong>void</strong></th>
<th><strong>reload</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reloads all material instances that use this material.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Point2F</strong></th>
<th><strong>rotPivotOffset</strong> [4]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The pivot position in UV coordinates to center the rotation animation.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong></th>
<th><strong>rotSpeed</strong> [4]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The speed to rotate the texture in degrees per second when rotation animation is enabled.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Point2F</strong></th>
<th><strong>scrollDir</strong> [4]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The scroll direction in UV space when scroll animation is enabled.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong></th>
<th><strong>scrollSpeed</strong> [4]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The speed to scroll the texture in UVs per second when scroll animation is enabled.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong></th>
<th><strong>sequenceFramePerSec</strong> [4]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The number of frames per second for frame based sequence animations if greater than zero.</td>
</tr>
<tr>
<td>Variable Type</td>
<td>Variable Name</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>float</code></td>
<td><code>sequenceSegmentSize</code> [4]</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>setAutoGenerated</code></td>
</tr>
<tr>
<td><code>filename</code></td>
<td><code>specularMap</code> [4]</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><code>subSurface</code> [4]</td>
</tr>
<tr>
<td><code>float</code></td>
<td><code>subSurfaceRolloff</code> [4]</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>bool translucent</code></td>
<td>If true this material is translucent blended.</td>
</tr>
<tr>
<td><code>MaterialBlendOp translucentBlendOp</code></td>
<td>The type of blend operation to use when the material is translucent.</td>
</tr>
<tr>
<td><code>bool translucentZWrite</code></td>
<td>If enabled and the material is translucent it will write into the depth buffer.</td>
</tr>
<tr>
<td><code>bool useAnisotropic [4]</code></td>
<td>Use anisotropic filtering for the textures of this stage.</td>
</tr>
<tr>
<td><code>bool vertColor [4]</code></td>
<td>If enabled, vertex colors are premultiplied with diffuse colors.</td>
</tr>
<tr>
<td><code>bool vertLit [4]</code></td>
<td>If true the vertex color is used for lighting.</td>
</tr>
<tr>
<td><code>float waveAmp [4]</code></td>
<td>The wave amplitude when wave animation is enabled.</td>
</tr>
<tr>
<td><code>float waveFreq [4]</code></td>
<td>The wave frequency when wave animation is enabled.</td>
</tr>
<tr>
<td><code>MaterialWaveType waveType [4]</code></td>
<td>The type of wave animation to perform when wave animation is enabled.</td>
</tr>
</tbody>
</table>

**Behavioral**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SFXTrack customFootstepSound</code></td>
<td>The sound to play when the player walks over the material. If this is set, it overrides <code>footstepSoundId</code>. This field is useful for directly assigning custom footstep sounds to materials without having to rely on the PlayerData sound</td>
</tr>
<tr>
<td>Type</td>
<td>Field</td>
</tr>
<tr>
<td>--------</td>
<td>----------------------</td>
</tr>
<tr>
<td>SFXTrack</td>
<td>customImpactSound</td>
</tr>
<tr>
<td>ColorF</td>
<td>effectColor [2]</td>
</tr>
<tr>
<td>int</td>
<td>footstepSoundId</td>
</tr>
<tr>
<td>int</td>
<td>impactSoundId</td>
</tr>
<tr>
<td>bool</td>
<td>showDust</td>
</tr>
<tr>
<td>bool</td>
<td>showFootprints</td>
</tr>
</tbody>
</table>
**Detailed Description**

A material in Torque 3D is a data structure that describes a surface. It contains many different types of information for rendering properties. Torque 3D generates shaders from Material definitions. The shaders are compiled at runtime and output into the example/shaders directory. Any errors or warnings generated from compiling the procedurally generated shaders are output to the console as well as the output window in the Visual C IDE.

**Example:**

```csharp
singleton Material (DECAL_scorch) {
    baseTex[0] = "./scorch_decal.png";
    vertColor[0] = true;

    translucent = true;
    translucentBlendOp = None;
    translucentZWrite = true;
    alphaTest = true;
    alphaRef = 84;
}
```

**See also:**

- Rendering
- ShaderData
### Member Data Documentation

**int** `Material::alphaRef`

The alpha reference value for alpha testing. Must be between 0 to 255.

**See also:**

`alphaTest`

**bool** `Material::alphaTest`

Enables alpha test when rendering the material.

**See also:**

`alphaRef`

**MaterialAnimType** `Material::animFlags[4]`

The types of animation to play on this material.

**filename** `Material::baseTex[4]`

For backwards compatibility.

**See also:**

`diffuseMap`

**bool** `Material::bumpAtlas[4]`

**filename** `Material::bumpTex[4]`
For backwards compatibility.

See also:

- normalMap

bool Material::castShadows

If set to false the lighting system will not cast shadows from this material.

Point2I Material::cellIndex[4]

Point2I Material::cellLayout[4]

int Material::cellSize[4]

ColorF Material::colorMultiply[4]

For backwards compatibility.

See also:

- diffuseColor

string Material::cubemap

The name of a CubemapData for environment mapping.

SFXTrack Material::customFootstepSound
The sound to play when the player walks over the material. If this is set, it overrides `footstepSoundId`. This field is useful for directly assigning custom footstep sounds to materials without having to rely on the `PlayerData` sound assignment.

Be aware that materials are client-side objects. This means that the SFXTracks assigned to materials must be client-side, too.

<table>
<thead>
<tr>
<th>SFXTrack Material::customImpactSound</th>
</tr>
</thead>
<tbody>
<tr>
<td>The sound to play when the player impacts on the surface with a velocity equal or greater than <code>PlayerData::groundImpactMinSpeed</code>. If this is set, it overrides <code>impactSoundId</code>. This field is useful for directly assigning custom impact sounds to materials without having to rely on the <code>PlayerData</code> sound assignment.</td>
</tr>
<tr>
<td>Be aware that materials are client-side objects. This means that the SFXTracks assigned to materials must be client-side, too.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>filename Material::detailMap[4]</th>
</tr>
</thead>
<tbody>
<tr>
<td>A typically greyscale detail texture additively blended into the material.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>filename Material::detailNormalMap[4]</th>
</tr>
</thead>
<tbody>
<tr>
<td>A second normal map texture applied at the detail scale. You can use the DXTnm format only when per-pixel specular highlights are disabled.</td>
</tr>
</tbody>
</table>

| float Material::detailNormalMapStrength[4] |
Used to scale the strength of the detail normal map when blended with the base normal map.

**Point2F** `Material::detailScale[4]`

The scale factor for the detail map.

**filename** `Material::detailTex[4]`

For backwards compatibility.

**See also:**

detailMap

**ColorF** `Material::diffuseColor[4]`

This color is multiplied against the diffuse texture color. If no diffuse texture is present this is the material color.

**filename** `Material::diffuseMap[4]`

The diffuse color texture map.

**bool** `Material::doubleSided`

Disables backface culling casing surfaces to be double sided. Note that the lighting on the backside will be a mirror of the front side of the surface.

**void** `Material::dumpInstances`
Dumps a formatted list of the currently allocated material instances for this material to the console.

```cpp
bool Material::dynamicCubemap
```

Enables the material to use the dynamic cubemap from the `ShapeBase` object its applied to.

```cpp
ColorF Material::effectColor[2]
```

If `showDust` is true, this is the set of colors to use for the `ParticleData` of the dust emitter.

**See also:**

```
ParticleData::colors
```

```cpp
bool Material::emissive[4]
```

Enables emissive lighting for the material.

```cpp
filename Material::envMap[4]
```

The name of an environment map cube map to apply to this material.

```cpp
filename Material::envTex[4]
```

For backwards compatibility.

**See also:**

```
envMap
```
**void** **Material::flush**

Flushes all material instances that use this material.

**int** **Material::footstepSoundId**

What sound to play from the **PlayerData** sound list when the player walks over the material. -1 (default) to not play any sound.

The IDs are:

- 0: **PlayerData::FootSoftSound**
- 1: **PlayerData::FootHardSound**
- 2: **PlayerData::FootMetalSound**
- 3: **PlayerData::FootSnowSound**
- 4: **PlayerData::FootShallowSound**
- 5: **PlayerData::FootWadingSound**
- 6: **PlayerData::FootUnderwaterSound**
- 7: **PlayerData::FootBubblesSound**
- 8: **PlayerData::movingBubblesSound**
- 9: **PlayerData::waterBreathSound**
- 10: **PlayerData::impactSoftSound**
- 11: **PlayerData::impactHardSound**
- 12: **PlayerData::impactMetalSound**
- 13: **PlayerData::impactSnowSound**
- 14: **PlayerData::impactWaterEasy**
- 15: **PlayerData::impactWaterMedium**
- 16: **PlayerData::impactWaterHard**
- 17: **PlayerData::exitingWater**

**string** **Material::getAnimFlags**

**string** **Material::getFilename**
Get filename of material.

<table>
<thead>
<tr>
<th>bool Material::glow[4]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enables rendering this material to the glow buffer.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int Material::impactSoundId</th>
</tr>
</thead>
<tbody>
<tr>
<td>What sound to play from the PlayerData sound list when the player impacts on the surface with a velocity equal or greater than PlayerData::groundImpactMinSpeed.</td>
</tr>
<tr>
<td>For a list of IDs, see footstepSoundId</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool Material::isAutoGenerated</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns true if this Material was automatically generated by MaterialList::mapMaterials().</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>filename Material::lightMap[4]</th>
</tr>
</thead>
<tbody>
<tr>
<td>The lightmap texture used with pureLight.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string Material::mapTo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Used to map this material to the material name used by TSShape.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float Material::minnaertConstant[4]</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Minnaert shading constant value. Must be greater than 0 to</td>
</tr>
</tbody>
</table>
enable the effect.

**filename Material::normalMap[4]**

The normal map texture. You can use the DXTnm format only when per-pixel specular highlights are disabled, or a specular map is in use.

**filename Material::overlayMap[4]**

A secondary diffuse color texture map which will use the second texcoord of a mesh.

**filename Material::overlayTex[4]**

For backwards compatibility.

**See also:**

overlayMap

**float Material::parallaxScale[4]**

Enables parallax mapping and defines the scale factor for the parallax effect. Typically this value is less than 0.4 else the effect breaks down.

**bool Material::pixelSpecular[4]**

This enables per-pixel specular highlights controlled by the alpha channel of the normal map texture. Note that if pixel specular is enabled the DXTnm format will not work with your normal map,
unless you are also using a specular map.

```cpp
bool Material::planarReflection
```

```cpp
void Material::reload
```

Reloads all material instances that use this material.

```cpp
Point2F Material::rotPivotOffset[4]
```

The pivot position in UV coordinates to center the rotation animation.

```cpp
float Material::rotSpeed[4]
```

The speed to rotate the texture in degrees per second when rotation animation is enabled.

```cpp
Point2F Material::scrollDir[4]
```

The scroll direction in UV space when scroll animation is enabled.

```cpp
float Material::scrollSpeed[4]
```

The speed to scroll the texture in UVs per second when scroll animation is enabled.

```cpp
float Material::sequenceFramePerSec[4]
```
The number of frames per second for frame based sequence animations if greater than zero.

```cpp
float Material::sequenceSegmentSize[4]
```

The size of each frame in UV units for sequence animations.

```cpp
void Material::setAutoGenerated
```

```cpp
setAutoGenerated(bool isAutoGenerated): Set whether or not the Material is autogenerated.
```

```cpp
bool Material::showDust
```

Whether to emit dust particles from a shape moving over the material. This is, for example, used by vehicles or players to decide whether to show dust trails.

```cpp
bool Material::showFootprints
```

Whether to show player footprint decals on this material.

See also:

```
PlayerData::decalData
```

```cpp
ColorF Material::specular[4]
```

The color of the specular highlight when not using a specularMap.
**filename Material::specularMap[4]**

The specular map texture. The RGB channels of this texture provide a per-pixel replacement for the 'specular' parameter on the material. If this texture contains alpha information, the alpha channel of the texture will be used as the gloss map. This provides a per-pixel replacement for the 'specularPower' on the material.

**float Material::specularPower[4]**

The intensity of the specular highlight when not using a specularMap.

**bool Material::subSurface[4]**

Enables the subsurface scattering approximation.

**ColorF Material::subSurfaceColor[4]**

The color used for the subsurface scattering approximation.

**float Material::subSurfaceRolloff[4]**

The 0 to 1 rolloff factor used in the subsurface scattering approximation.

**filename Material::toneMap[4]**

The tonemap texture used with pureLight.
**bool Material::translucent**

If true this material is translucent blended.

**MaterialBlendOp Material::translucentBlendOp**

The type of blend operation to use when the material is translucent.

**bool Material::translucentZWrite**

If enabled and the material is translucent it will write into the depth buffer.

**bool Material::useAnisotropic[4]**

Use anisotropic filtering for the textures of this stage.

**bool Material::vertColor[4]**

If enabled, vertex colors are premultiplied with diffuse colors.

**bool Material::vertLit[4]**

If true the vertex color is used for lighting.

**float Material::waveAmp[4]**

The wave amplitude when wave animation is enabled.
float Material::waveFreq[4]

The wave frequency when wave animation is enabled.

MaterialWaveType Material::waveType[4]

The type of wave animation to perform when wave animation is enabled.
MeshRoad Class Reference
[Terrain]

A strip of rectangular mesh segments defined by a 3D spline for prototyping road-shaped objects in your scene. More...

Inheritance diagram for MeshRoad:

```
+----------------+   +----------------+   +----------------+   +----------------+
|     SimObject  |   |      NetObject  |   | SceneObject     |   |  MeshRoad       |
|       |       |               |   |               |   |               |
|       +-------+   +-------+   +-------+   +-------+
|       |       |               |   |       |               |   |               |
|       |       |       |               |   |       |               |   |               |
|       |       |       |               |   |       |               |   |               |
|       |       |       |               |   |       |               |   |               |
|       |       |       |               |   |       |               |   |               |
|       |       |       |               |   |       |               |   |               |
|       |       |       |               |   |       |               |   |               |
|       |       |       |               |   |       |               |   |               |
|       |       |       |               |   |       |               |   |               |
```
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void postApply ()</code></td>
<td>Intended as a helper to developers and editor scripts.</td>
</tr>
<tr>
<td><code>void regenerate ()</code></td>
<td>Intended as a helper to developers and editor scripts.</td>
</tr>
<tr>
<td><code>void setNodeDepth (int idx, float meters)</code></td>
<td>Intended as a helper to developers and editor scripts.</td>
</tr>
</tbody>
</table>
## Public Attributes

### MeshRoad

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td><strong>bottomMaterial</strong></td>
<td>Material for the bottom surface of the road.</td>
</tr>
<tr>
<td>float</td>
<td><strong>breakAngle</strong></td>
<td>Angle in degrees - <strong>MeshRoad</strong> will subdivide the spline if its curve is greater than this threshold.</td>
</tr>
<tr>
<td>string</td>
<td><strong>sideMaterial</strong></td>
<td>Material for the left, right, front, and back surfaces of the road.</td>
</tr>
<tr>
<td>float</td>
<td><strong>textureLength</strong></td>
<td>The length in meters of textures mapped to the <strong>MeshRoad</strong>.</td>
</tr>
<tr>
<td>string</td>
<td><strong>topMaterial</strong></td>
<td>Material for the upper surface of the road.</td>
</tr>
<tr>
<td>int</td>
<td><strong>widthSubdivisions</strong></td>
<td>Subdivide segments widthwise this many times when generating vertices.</td>
</tr>
</tbody>
</table>

## Internal

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td><strong>Node</strong></td>
<td>Do not modify, for internal use.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td><strong>EditorOpen</strong></td>
<td>True if the <code>MeshRoad</code> editor is open, otherwise false.</td>
</tr>
<tr>
<td>static bool</td>
<td><strong>isRenderable</strong></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td><strong>isSelectable</strong></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td><strong>showBatches</strong></td>
<td>Determines if the debug rendering of the batches cubes is displayed or not.</td>
</tr>
<tr>
<td>static bool</td>
<td><strong>showRoad</strong></td>
<td>If true, the road will be rendered. When in the editor, roads are always rendered regardless of this flag.</td>
</tr>
<tr>
<td>static bool</td>
<td><strong>showSpline</strong></td>
<td>If true, the spline on which the curvature of this road is based will be rendered.</td>
</tr>
<tr>
<td>static bool</td>
<td><strong>wireframe</strong></td>
<td>If true, will render the wireframe of the road.</td>
</tr>
</tbody>
</table>
Detailed Description

A strip of rectangular mesh segments defined by a 3D spline for prototyping road-shaped objects in your scene.

User may control width and depth per node, overall spline shape in three dimensions, and separate Materials for rendering the top, bottom, and side surfaces.

MeshRoad is not capable of handling intersections, branches, curbs, or other desirable features in a final 'road' asset and is therefore intended for prototyping and experimentation.

Materials assigned to MeshRoad should tile vertically.
Member Function Documentation

void MeshRoad::postApply(
)

Intended as a helper to developers and editor scripts.
Force trigger an inspectPostApply. This will transmit material and other fields (not including nodes) to client objects.

void MeshRoad::regenerate(
)

Intended as a helper to developers and editor scripts.
Force MeshRoad to recreate its geometry.

void MeshRoad::setNodeDepth(int idx, float meters)

Intended as a helper to developers and editor scripts.
Sets the depth in meters of a particular node.
Member Data Documentation

string MeshRoad::bottomMaterial

Material for the bottom surface of the road.

float MeshRoad::breakAngle

Angle in degrees - MeshRoad will subdivide the spline if its curve is greater than this threshold.

bool MeshRoad::EditorOpen [static]

True if the MeshRoad editor is open, otherwise false.

string MeshRoad::Node

Do not modify, for internal use.

bool MeshRoad::showBatches [static]

Determines if the debug rendering of the batches cubes is displayed or not.

bool MeshRoad::showRoad [static]

If true, the road will be rendered. When in the editor, roads are always rendered regardless of this flag.
bool MeshRoad::showSpline [static]

If true, the spline on which the curvature of this road is based will be rendered.

string MeshRoad::sideMaterial

Material for the left, right, front, and back surfaces of the road.

float MeshRoad::textureLength

The length in meters of textures mapped to the MeshRoad.

string MeshRoad::topMaterial

Material for the upper surface of the road.

int MeshRoad::widthSubdivisions

Subdivide segments widthwise this many times when generating vertices.

bool MeshRoad::wireframe [static]

If true, will render the wireframe of the road.
Message Class Reference
[Messaging]

Base class for messages. More...

Inheritance diagram for Message:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>void</th>
<th>addReference ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Increment the reference count for this message.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>freeReference ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Decrement the reference count for this message.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>getType ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Get message type (script class name or C++ class name if no script defined class).</td>
</tr>
</tbody>
</table>

## Callbacks

<table>
<thead>
<tr>
<th>void</th>
<th>onAdd ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Script callback when a message is first created and registered.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>onRemove ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Script callback when a message is deleted.</td>
</tr>
</tbody>
</table>
Detailed Description

Base class for messages.

Message is the base class for C++ defined messages, and may also be used in script for script defined messages if no C++ subclass is appropriate.

Messages are reference counted and will be automatically deleted when their reference count reaches zero. When you dispatch a message, a reference will be added before the dispatch and freed after the dispatch. This allows for temporary messages with no additional code. If you want to keep the message around, for example to dispatch it to multiple queues, call addReference() before dispatching it and freeReference() when you are done with it. Never delete a Message object directly unless addReference() has not been called or the message has not been dispatched.

Message IDs are pooled similarly to datablocks, with the exception that IDs are reused. If you keep a message for longer than a single dispatch, then you should ensure that you clear any script variables that refer to it after the last freeReference(). If you don't, then it is probable that the object ID will become valid again in the future and could cause hard to track down bugs.

Messages have a unique type to simplify message handling code. For object messages, the type is defined as either the script defined class name or the C++ class name if no script class was defined. The message type may be obtained through the getType() method.

By convention, any data for the message is held in script accessible fields. Messages that need to be handled in C++ as well as script provide the relevant data through persistent fields in a subclass of Message to provide best performance on the C++ side. Script defined messages usually their through dynamic fields, and may be accessed in C++ using the SimObject::getDataField() method.
Member Function Documentation

void Message::addReference( )
Increment the reference count for this message.

void Message::freeReference( )
Decrement the reference count for this message.

string Message::getType( )
Get message type (script class name or C++ class name if no script defined class).

void Message::onAdd( )
Script callback when a message is first created and registered.

Example:

function Message::onAdd(%this)
{
    // Perform on add code here
}

void Message::onRemove( )
Script callback when a message is deleted.

Example:
function Message::onRemove(%this) {
    // Perform on remove code here
}

Copyright © GarageGames, LLC. All Rights Reserved.
MessageForwarder Class Reference
[Messaging]

Forward messages from one queue to another. More...

Inheritance diagram for MessageForwarder:

[SimObject]

[ScriptMsgListener]

[MessageForwarder]

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>caseString</th>
<th>toQueue</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Name of queue to forward to.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Forward messages from one queue to another.

**MessageForwarder** is a script class that can be used to forward messages from one queue to another.

**Example:**

```java
%fwd = new MessageForwarder()
{
    toQueue = "QueueToSendTo";
};

registerMessageListener("FromQueue", %fwd)
```

Where "QueueToSendTo" is the queue you want to forward to, and "FromQueue" is the queue you want to forward from.
Member Data Documentation

`caseString MessageForwarder::toQueue`

Name of queue to forward to.
MessageVector Class Reference

[Utility Controls]

Store a list of chat messages. More...

Inheritance diagram for MessageVector:

List of all members.
**Public Member Functions**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void clear ()</td>
<td>Clear all messages in the vector.</td>
</tr>
<tr>
<td>bool deleteLine (int deletePos)</td>
<td>Delete the line at the specified position.</td>
</tr>
<tr>
<td>void dump (string filename)</td>
<td>Dump the message vector to a file without a header.</td>
</tr>
<tr>
<td>void dump (string filename, string header)</td>
<td>Dump the message vector to a file with a header.</td>
</tr>
<tr>
<td>int getlineIndexByTag (int tag)</td>
<td>Scan through the vector, returning the line number of the first line that matches the specified tag; else returns -1 if no match was found.</td>
</tr>
<tr>
<td>int getlineTag (int pos)</td>
<td>Get the tag of a specified line.</td>
</tr>
<tr>
<td>string getlineText (int pos)</td>
<td>Get the text at a specified line.</td>
</tr>
<tr>
<td>string getlineTextByTag (int tag)</td>
<td>Scan through the lines in the vector, returning the first line that has a matching tag.</td>
</tr>
<tr>
<td>int getNumLines ()</td>
<td>Get the number of lines in the vector.</td>
</tr>
<tr>
<td>bool insertLine (int insertPos, string msg, int tag)</td>
<td>Push a line onto the back of the list.</td>
</tr>
<tr>
<td>bool popBackLine ()</td>
<td>Pop a line from the back of the list; destroys the line.</td>
</tr>
<tr>
<td>bool popFrontLine ()</td>
<td>Pop a line from the front of the vector, destroying the line.</td>
</tr>
<tr>
<td>void pushBackLine (string msg, int tag)</td>
<td>Push a line onto the back of the list.</td>
</tr>
</tbody>
</table>
void pushFrontLine (string msg, int tag)
  Push a line onto the front of the vector.
Detailed Description

Store a list of chat messages.

This is responsible for managing messages which appear in the chat HUD, not the actual control rendered to the screen

Example:

```cpp
// Declare ChatHud, which is what will display the actual chat from MessageVector
new GuiMessageVectorCtrl(ChatHud) {
    profile = "ChatHudMessageProfile";
    horizSizing = "width";
    vertSizing = "height";
    position = "1 1";
    extent = "252 16";
    minExtent = "8 8";
    visible = "1";
    helpTag = "0";
    lineSpacing = "0";
    lineContinuedIndex = "10";
    matchColor = "0 0 255 255";
    maxColorIndex = "5";
};

// All messages are stored in this HudMessageVector, the actual MainChatHud only displays the contents
new MessageVector(HudMessageVector);

// Attach the MessageVector to the chat control
chatHud.attach(HudMessageVector);
```

See also:  

`GuiMessageVectorCtrl` for more details on how this is used.
**Member Function Documentation**

**void MessageVector::clear( )**

Clear all messages in the vector.

**Example:**

```
HudMessageVector.clear();
```

**bool MessageVector::deleteLine(int deletePos )**

Delete the line at the specified position.

**Parameters:**

- `deletePos` Position in the vector containing the line to be deleted

**Example:**

```
// Delete the first line (index 0) in the
HudMessageVector.deleteLine(0);
```

**Returns:**

False if `deletePos` is greater than the number of lines in the current vector

**void MessageVector::dump(string filename )**

Dump the message vector to a file without a header.

**Parameters:**

- `filename` Name and path of file to dump text to.
Example:

```cpp
// Dump the entire chat log to a text file
HudMessageVector.dump("./chatLog.txt");
```

```cpp
void MessageVector::dump(string filename,
                         string header
)
```

Dump the message vector to a file with a header.

**Parameters:**

- `filename` Name and path of file to dump text to.
- `header` Prefix information for write out

**Example:**

```cpp
// Arbitrary header data
%headerInfo = "Ars Moriendi Chat Log";

// Dump the entire chat log to a text file
HudMessageVector.dump("./chatLog.txt", %headerInfo);
```

```cpp
int MessageVector::getLineIndexByTag(int tag )
```

Scan through the vector, returning the line number of the first line that matches the specified tag; else returns -1 if no match was found.

**Parameters:**

- `tag` Numerical value assigned to a message when it was added or inserted

**Example:**
// Locate a line of text tagged with the value 1, then delete it.
%taggedLine = HudMessageVector.getLineIndexByTag(1);
HudMessageVector.deleteLine(%taggedLine);

Returns:
Line with matching tag, otherwise -1

int MessageVector::getLineTag(int pos )

Get the tag of a specified line.

Parameters:
pos Position in vector to grab tag from

Example:

// Remove all lines that do not have a tag
while( HudMessageVector.getNumLines())
{
   %tag = HudMessageVector.getLineTag(1);
   if(%tag != 1)
      %tag.delete();
   HudMessageVector.popFrontLine();
}

Returns:
Tag value of a given line, if the position is greater than the number of lines return 0

string MessageVector::getLineText(int pos )

Get the text at a specified line.
**Parameters:**

*pos* Position in vector to grab text from

**Example:**

```plaintext
// Print a line of text at position 1.
%text = HudMessageVector.getLineText(1);
echo(%text);
```

**Returns:**
Text at specified line, if the position is greater than the number of lines return ""

```plaintext
string MessageVector::getLineTextByTag(int tag )
```

Scan through the lines in the vector, returning the first line that has a matching tag.

**Parameters:**

*tag* Numerical value assigned to a message when it was added or inserted

**Example:**

```plaintext
// Locate text in the vector tagged with 1
%taggedText = HudMessageVector.getLineTextByTag(1);
echo(%taggedText);
```

**Returns:**
Text from a line with matching tag, other wise ""

```plaintext
int MessageVector::getNumLines( )
```

Get the number of lines in the vector.
Example:

```c++
// Find out how many lines have been stored
%chatLines = HudMessageVector.getNumLines();
echo(%chatLines);
```

```c++
bool MessageVector::insertLine(int insertPos, string msg, int tag)
```

Push a line onto the back of the list.

**Parameters:**

- `msg` Text that makes up the message
- `tag` Numerical value associated with this message, useful for searching.

Example:

```c++
// Add the message...
HudMessageVector.insertLine(1, "Hello World");
```

**Returns:**

False if `insertPos` is greater than the number of lines in the current vector

```c++
bool MessageVector::popBackLine()
```

Pop a line from the back of the list; destroys the line.

Example:

```c++
HudMessageVector.popBackLine();
```
Returns:
False if there are no lines to pop (underflow), true otherwise

```cpp
bool MessageVector::popFrontLine() {
    Pop a line from the front of the vector, destroying the line.
    Example:
    HudMessageVector.popFrontLine();
    Returns:
    False if there are no lines to pop (underflow), true otherwise
}
```

```cpp
void MessageVector::pushBackLine(string msg, int tag) {
    Push a line onto the back of the list.
    Parameters:
    msg Text that makes up the message
    tag Numerical value associated with this message, useful for searching.
    Example:
    // Add the message...
    HudMessageVector.pushBackLine("Hello World");
}
```

```cpp
void MessageVector::pushFrontLine(string msg, int tag) {
    // Function body
}
```
Push a line onto the front of the vector.

**Parameters:**

- `msg` Text that makes up the message
- `tag` Numerical value associated with this message, useful for searching.

**Example:**

```cpp
// Add the message...
HudMessageVector.pushFrontLine("Hello World");
```

Copyright © GarageGames, LLC. All Rights Reserved.
MissionArea Class Reference
[ atención ]

Level object which defines the boundaries of the level. More...

Inheritance diagram for MissionArea:

```
SimObject

NetObject

MissionArea

[legend]
```

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>String</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td><strong>getArea</strong> ()</td>
<td>Returns 4 fields: starting x, starting y, extents x, extents y.</td>
</tr>
<tr>
<td>void</td>
<td><strong>postApply</strong> ()</td>
<td>Intended as a helper to developers and editor scripts.</td>
</tr>
<tr>
<td>void</td>
<td><strong>setArea</strong> (int x, int y, int width, int height)</td>
<td>Defines the size of the MissionArea</td>
</tr>
</tbody>
</table>
## Public Attributes

### Dimensions

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RectI</td>
<td>area</td>
<td>Four corners (X1, X2, Y1, Y2) that makes up the level's boundaries.</td>
</tr>
<tr>
<td>float</td>
<td>flightCeiling</td>
<td>Represents the top of the mission area, used by FlyingVehicle.</td>
</tr>
<tr>
<td>float</td>
<td>flightCeilingRange</td>
<td>Distance from ceiling before FlyingVehicle thrust is cut off.</td>
</tr>
</tbody>
</table>
Detailed Description

Level object which defines the boundaries of the level.

This is a simple box with starting points, width, depth, and height. It does not have any default functionality. Instead, when objects hit the boundaries certain script callbacks will be made allowing you to control the reaction.

Example:

```plaintext
new MissionArea(GlobalMissionArea)
{
    Area = "-152 -352 1008 864";
    flightCeiling = "300";
    flightCeilingRange = "20";
    canSaveDynamicFields = "1";
    enabled = "1";
    TypeBool locked = "false";
};
```
Member Function Documentation

string MissionArea::getArea()

Returns 4 fields: starting x, starting y, extents x, extents y.

void MissionArea::postApply()

Intended as a helper to developers and editor scripts.

Force trigger an inspectPostApply. This will transmit material and other fields (not including nodes) to client objects.

void MissionArea::setArea(int x, int y, int width, int height)

- Defines the size of the MissionArea

param x Starting X coordinate position for MissionArea
param y Starting Y coordinate position for MissionArea
param width New width of the MissionArea
param height New height of the MissionArea

Note:

Only the server object may be set.
Member Data Documentation

RectI MissionArea::area

Four corners (X1, X2, Y1, Y2) that makes up the level's boundaries.

float MissionArea::flightCeiling

Represents the top of the mission area, used by FlyingVehicle.

float MissionArea::flightCeilingRange

Distance from ceiling before FlyingVehicle thrust is cut off.

Copyright © GarageGames, LLC. All Rights Reserved.
MissionMarker Class Reference
[Miscellaneous]

This is a base class for all "marker" related objects. It is a 3D representation of a point in the level. More...

Inheritance diagram for MissionMarker:

```
    SimObject
     ↓    ↓
   NetObject
     ↓    ↓
 SceneObject
     ↓    ↓
 GameBase
     ↓    ↓
 ShapeBase
     ↓    ↓
 MissionMarker
     ↓    ↓
 SpawnSphere WayPoint
```

List of all members.
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

This is a base class for all "marker" related objects. It is a 3D representation of a point in the level.

The main use of a MissionMarker is to represent a point in 3D space with a mesh and basic ShapeBase information. If you simply need to mark a spot in your level, with no overhead from additional fields, this is a useful object.

Example:

```java
new MissionMarker()
{
    dataBlock = "WayPointMarker";
    position = "295.699 -171.817 280.124";
    rotation = "0 0 -1 13.8204";
    scale = "1 1 1";
    isRenderEnabled = "true";
    canSaveDynamicFields = "1";
    enabled = "1";
};
```

Note:

MissionMarkers will not add themselves to the scene except when in the editor.

See also:

MissionMarkerData
SpawnSphere
WayPoint

Copyright © GarageGames, LLC. All Rights Reserved.
MissionMarkerData Class Reference

[Miscellaneous]

A very basic class containing information used by MissionMarker objects for rendering. More...

Inheritance diagram for MissionMarkerData:

[diagram]

List of all members.
Detailed Description

A very basic class containing information used by MissionMarker objects for rendering.

MissionMarkerData, is an extremely barebones class derived from ShapeBaseData. It is solely used by MissionMarker classes (such as SpawnSphere), so that you can see the object while editing a level.

Example:

datablock MissionMarkerData(SpawnSphereMarker) {
    category = "Misc";
    shapeFile = "core/art/shapes/octahedron"
};

See also:

MissionMarker
SpawnSphere
WayPoint

Copyright © GarageGames, LLC. All Rights Reserved.
NetConnection Class Reference
[Networking]

Provides the basis for implementing a multiplayer game protocol. More...

Inheritance diagram for NetConnection:

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void checkMaxRate ()</td>
<td>Ensures that all configured packet rates and sizes meet minimum requirements.</td>
</tr>
<tr>
<td>void clearPaths ()</td>
<td>On the server, resets the connection to indicate that motion spline paths have not been transmitted.</td>
</tr>
<tr>
<td>void connect (string remoteAddress)</td>
<td>Connects to the remote address.</td>
</tr>
<tr>
<td>string connectLocal ()</td>
<td>Connects with the server that is running within the same process as the client.</td>
</tr>
<tr>
<td>string getAddress ()</td>
<td>Returns the far end network address for the connection.</td>
</tr>
<tr>
<td>int getGhostID (int realID)</td>
<td>On server or client, convert a real id to the ghost id for this connection.</td>
</tr>
<tr>
<td>int getGhostsActive ()</td>
<td>Provides the number of active ghosts on the connection.</td>
</tr>
<tr>
<td>int getPacketLoss ()</td>
<td>Returns the percentage of packets lost per tick.</td>
</tr>
<tr>
<td>int getPing ()</td>
<td>Returns the average round trip time (in ms) for the connection.</td>
</tr>
<tr>
<td>int resolveGhostID (int ghostID)</td>
<td>On the client, convert a ghost ID from this connection to a real SimObject ID.</td>
</tr>
<tr>
<td>int resolveObjectFromGhostIndex (int ghostID)</td>
<td>On the server, convert a ghost ID from this connection to a real SimObject ID.</td>
</tr>
<tr>
<td>void setSimulatedNetParams (float packetLoss, int delay)</td>
<td></td>
</tr>
</tbody>
</table>
Simulate network issues on the connection for testing.

```c
void transmitPaths ()
    Sent by the server during phase 2 of the mission download to update motion spline paths.
```
Detailed Description

Provides the basis for implementing a multiplayer game protocol.

**NetConnection** combines a low-level notify protocol implemented in ConnectionProtocol with a **SimGroup**, and implements several distinct subsystems:

- **Event Manager** This is responsible for transmitting NetEvents over the wire. It deals with ensuring that the various types of NetEvents are delivered appropriately, and with notifying the event of its delivery status.

- **Move Manager** This is responsible for transferring a Move to the server 32 times a second (on the client) and applying it to the control object (on the server).

- **Ghost Manager** This is responsible for doing scoping calculations (on the server side) and transmitting most-recent ghost information to the client.

- **File Transfer** It is often the case that clients will lack important files when connecting to a server which is running a mod or new map. This subsystem allows the server to transfer such files to the client.

- **Networked String Table** String data can easily soak up network bandwidth, so for efficiency, we implement a networked string table. We can then notify the connection of strings we will reference often, such as player names, and transmit only a tag, instead of the whole string.

- **Demo Recording** A demo in Torque is a log of the network traffic between client and server; when a **NetConnection** records a demo, it simply logs this data to a file. When it plays a demo back, it replays the logged data.

- **Connection Database** This is used to keep track of all the NetConnections; it can be iterated over (for instance, to send an
event to all active connections), or queried by address.

The **NetConnection** is a **SimGroup**. On the client side, it contains all the objects which have been ghosted to that client. On the server side, it is empty; it can be used (typically in script) to hold objects related to the connection. For instance, you might place an observation camera in the NetConnection. In both cases, when the connection is destroyed, so are the contained objects.

The **NetConnection** also has the concept of local connections. These are used when the client and server reside in the same process. A local connection is typically required to use the standard Torque world building tools. A local connection is also required when building a single player game.

**See also:**

- Networking, On Ghosting and Scoping, NetConnection Group, Local Connections, GameConnection, AIConnection, and AIClient.
**Member Function Documentation**

### void NetConnection::checkMaxRate()

Ensures that all configured packet rates and sizes meet minimum requirements.

This method is normally only called when a `NetConnection` class is first constructed. It need only be manually called if the global variables that set the packet rate or size have changed.

**Note:**

If `$pref::Net::PacketRateToServer`, `$pref::Net::PacketRateToClient` or `$pref::Net::PacketSize` have been changed since a `NetConnection` has been created, this method must be called on all connections for them to follow the new rates or size.

### void NetConnection::clearPaths()

On the server, resets the connection to indicate that motion spline paths have not been transmitted.

Typically when a mission has ended on the server, all connected clients are informed of this change and their connections are reset back to a starting state. This method resets a connection on the server to indicate that motion spline paths have not been transmitted.

**Example:**

```c++
// Inform the clients
for (%clientIndex = 0; %clientIndex < (;
{
    // clear ghosts and paths from all clients
    %cl = ClientGroup.getObject(%clientIndex);
    %cl.clearGhostPaths();
}
```
%cl.endMission();
%cl.resetGhosting();
%cl.clearPaths();
}

See also:
transmitPaths()
Path

```c++
void NetConnection::connect(string remoteAddress )
```

Connects to the remote address.

Attempts to connect with another NetConnection on the given address. Typically once connected, a game's information is passed along from the server to the client, followed by the player entering the game world. The actual procedure is dependent on the NetConnection subclass that is used. i.e. GameConnection.

**Parameters:**

- `remoteAddress` The address to connect to in the form of IP: <address>:<port although the IP: portion is optional. The address portion may be in the form of w.x.y.z or as a host name, in which case a DNS lookup will be performed. You may also substitute the word broadcast for the address to broadcast the connect request over the local subnet.

**See also:**

- NetConnection::connectLocal() to connect to a server running within the same process as the client.

```c++
string NetConnection::connectLocal( )
```
Connects with the server that is running within the same process as the client.

**Returns:**

An error text message upon failure, or an empty string when successful.

**See also:**

See Local Connections for a description of local connections and their use. See NetConnection::connect() to connect to a server running in another process (on the same machine or not).

```cpp
string NetConnection::getAddress()
```

Returns the far end network address for the connection.

The address will be in one of the following forms:

- **IP:Broadcast:<port>** for broadcast type addresses
- **IP:<address>:<port>** for IP addresses
- **local** when connected locally (server and client running in same process)

```cpp
int NetConnection::getGhostID(int realID)
```

On server or client, convert a real id to the ghost id for this connection.

Torque's network ghosting system only exchanges ghost ID's between the server and client. Use this method on the server or client to discover an object's ghost ID based on its real SimObject ID.

**Parameters:**

- `realID` The real SimObject ID of the object.
Returns:
The ghost ID of the object for this connection, or -1 if it could not be resolved.

See also:
On Ghosting and Scoping for a description of the ghosting system.

```
int NetConnection::getGhostsActive()
```

Provides the number of active ghosts on the connection.

Returns:
The number of active ghosts.

See also:
On Ghosting and Scoping for a description of the ghosting system.

```
int NetConnection::getPacketLoss()
```

Returns the percentage of packets lost per tick.

Note:
This method is not yet hooked up.

```
int NetConnection::getPing()
```

Returns the average round trip time (in ms) for the connection.

The round trip time is recalculated every time a notify packet is received. Notify packets are used to information the connection that the far end successfully received the sent packet.
int NetConnection::resolveGhostID(int ghostID)

On the client, convert a ghost ID from this connection to a real SimObject ID.

Torque's network ghosting system only exchanges ghost ID's between the server and client. Use this method on the client to discover an object's local SimObject ID when you only have a ghost ID.

**Parameters:**

ghostID The ghost ID of the object as sent by the server.

**Returns:**

The SimObject ID of the object, or 0 if it could not be resolved.

**Example:**

```
%object = ServerConnection.resolveGhostID(%ghostId);
```

**See also:**

On Ghosting and Scoping for a description of the ghosting system.

int NetConnection::resolveObjectFromGhostIndex(int ghostID)

On the server, convert a ghost ID from this connection to a real SimObject ID.

Torque's network ghosting system only exchanges ghost ID's between the server and client. Use this method on the server to discover an object's local SimObject ID when you only have a ghost ID.

**Parameters:**
ghostID  The ghost ID of the object as sent by the server.

Returns:

The SimObject ID of the object, or 0 if it could not be resolved.

Example:

```%object = %client.resolveObjectFromGhostIndex(%ghostId);```

See also:

On Ghosting and Scoping for a description of the ghosting system.

```void NetConnection::setSimulatedNetParams(float packetLoss, int delay)```

Simulate network issues on the connection for testing.

Parameters:

- `packetLoss`  The fraction of packets that will be lost. Ranges from 0.0 (no loss) to 1.0 (complete loss)
- `delay`  Delays packets being transmitted by simulating a particular ping. This is an absolute integer, measured in ms.

```void NetConnection::transmitPaths( )```

Sent by the server during phase 2 of the mission download to update motion spline paths.

The server transmits all spline motion paths that are within the mission (Path) separate from other objects. This is due to the potentially large number of nodes within each path, which may saturate a packet sent to the client. By managing this step
separately, Torque has finer control over how packets are organised vs. doing it during the ghosting stage.

Internally a PathManager is used to track all paths defined within a mission on the server, and each one is transmitted using a PathManagerEvent. The client side collects these events and builds the given paths within its own PathManager. This is typically done during the standard mission start phase 2 when following Torque's example mission startup sequence.

When a mission is ended, all paths need to be cleared from their respective path managers.

Example:

```lisp
function serverCmdMissionStartPhase2Ack(%client, %seq, %playerDB)
{
    // Make sure to ignore calls from a previous mission load
    if (%seq != $missionSequence || !$MissionRunning)
        return;

    if (%client.currentPhase != 1.5)
        return;
    %client.currentPhase = 2;

    // Set the player datablock choice
    %client.playerDB = %playerDB;

    // Update mission paths (SimPath), this needs to get there before the objects
    %client.transmitPaths();

    // Start ghosting objects to the client
    %client.activateGhosting();
}
```

See also:

clearPaths()
NetObject Class Reference
[Networking]

Superclass for all ghostable networked objects. More...

Inheritance diagram for NetObject:
List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void clearScopeToClient (NetConnection client)</td>
<td>Undo the effects of a <code>scopeToClient()</code> call.</td>
</tr>
<tr>
<td>int getClientObject ()</td>
<td>Returns a pointer to the client object when on a local connection.</td>
</tr>
<tr>
<td>int getGhostID ()</td>
<td>Get the ghost index of this object from the server.</td>
</tr>
<tr>
<td>int getServerObject ()</td>
<td>Returns a pointer to the client object when on a local connection.</td>
</tr>
<tr>
<td>bool isClientObject ()</td>
<td>Called to check if an object resides on the clientside.</td>
</tr>
<tr>
<td>bool isServerObject ()</td>
<td>Checks if an object resides on the server.</td>
</tr>
<tr>
<td>void scopeToClient (NetConnection client)</td>
<td>Cause the <code>NetObject</code> to be forced as scoped on the specified <code>NetConnection</code>.</td>
</tr>
<tr>
<td>void setScopeAlways ()</td>
<td>Always scope this object on all connections.</td>
</tr>
</tbody>
</table>
Detailed Description

Superclass for all ghostable networked objects.
Introduction To NetObject And Ghosting

This class is the basis of the ghost implementation in Torque 3D. Every 3D object is a NetObject. One of the most powerful aspects of Torque's networking code is its support for ghosting and prioritized, most-recent-state network updates. The way this works is a bit complex, but it is immensely efficient. Let's run through the steps that the server goes through for each client in this part of Torque's networking:

- First, the server determines what objects are in-scope for the client. This is done by calling onCameraScopeQuery() on the object which is considered the "scope" object. This is usually the player object, but it can be something else. (For instance, the current vehicle, or a object we're remote controlling.)

- Second, it ghosts them to the client; this is implemented in netGhost.cc.

- Finally, it sends updates as needed, by checking the dirty list and packing updates.

There several significant advantages to using this networking system:

- Efficient network usage, since we only send data that has changed. In addition, since we only care about most-recent data, if a packet is dropped, we don't waste effort trying to deliver stale data.

- Cheating protection; since we don't deliver information about game objects which aren't in scope, we dramatically reduce the ability of clients to hack the game and gain a meaningful advantage. (For instance, they can't find out about things behind them, since objects behind them don't fall in scope.) In addition, since ghost IDs are assigned per-client, it's difficult for any sort of co-ordination between cheaters to occur.

NetConnection contains the Ghost Manager implementation, which
deals with transferring data to the appropriate clients and keeping state in synch.

See also:

On Ghosting and Scoping for a description of the ghosting system.
An Example Implementation

The basis of the ghost implementation in Torque is NetObject. It tracks the dirty flags for the various states that the object wants to network, and does some other book-keeping to allow more efficient operation of the networking layer.

Using a NetObject is very simple; let's go through a simple example implementation:

```cpp
class SimpleNetObject : public NetObject
{
public:
    typedef NetObject Parent;
    DECLARE_CONOBJECT(SimpleNetObject);
}
```

Above is the standard boilerplate code for a Torque class. You can find out more about this in SimObject.

```cpp
char message1[256];
char message2[256];
enum States {
    Message1Mask = BIT(0),
    Message2Mask = BIT(1),
};
```

For our example, we're having two "states" that we keep track of, message1 and message2. In a real object, we might map our states to health and position, or some other set of fields. You have 32 bits to work with, so it's possible to be very specific when defining states. In general, you should try to use as few states as possible (you never know when you'll need to expand your object's functionality!), and in fact, most of your fields will end up changing all at once, so it's not worth it to be too fine-grained. (As an example, position and velocity on Player are controlled by the same bit, as one rarely changes without the other changing, too.)
SimpleNetObject()
{
    // In order for an object to be considered by the network system,
    // the Ghostable net flag must be set.
    // The ScopeAlways flag indicates that the object is always scoped
    // on all active connections.
    mNetFlags.set(ScopeAlways | Ghostable);
    dStrcpy(message1, "Hello World 1!");
    dStrcpy(message2, "Hello World 2!");
}

Here is the constructor. Here, you see that we initialize our net flags
to show that we should always be scoped, and that we're to be taken
into consideration for ghosting. We also provide some initial values
for the message fields.

U32 packUpdate(NetConnection *, U32 mask)
{
    // check which states need to be updated
    if(stream->writeFlag(mas & Message1Mask))
        stream->writeString(message1);
    if(stream->writeFlag(mas & Message2Mask))
        stream->writeString(message2);

    // the return value from packUpdate
    // need to be updated for this object
    return 0;
}

Here's half of the meat of the networking code, the packUpdate() function. (The other half, unpackUpdate(), we'll get to in a second.)
The comments in the code pretty much explain everything, however,
notice that the code follows a pattern of if(writeFlag(mas &
The packUpdate() / unpackUpdate() functions are responsible for reading and writing the dirty bits to the bitstream by themselves.

```cpp
void unpackUpdate(NetConnection *, BitStream * stream) {
    // the unpackUpdate function must be symmetrical to packUpdate
    if (stream->readFlag()) {
        stream->readString(message1);
        Con::printf("Got message1: %s", message1);
    }
    if (stream->readFlag()) {
        stream->readString(message2);
        Con::printf("Got message2: %s", message2);
    }
}
```

The other half of the networking code in any NetObject, unpackUpdate(). In our simple example, all that the code does is print the new messages to the console; however, in a more advanced object, you might trigger animations, update complex object properties, or even spawn new objects, based on what packet data you unpack.

```cpp
void setMessage1(const char * msg) {
    setMaskBits(Message1Mask);
    dStrcpy(message1, msg);
}
void setMessage2(const char * msg) {
    setMaskBits(Message2Mask);
}
```
Here are the accessors for the two properties. It is good to encapsulate your state variables, so that you don't have to remember to make a call to setMaskBits every time you change anything; the accessors can do it for you. In a more complex object, you might need to set multiple mask bits when you change something; this can be done using the | operator, for instance, setMaskBits( Message1Mask | Message2Mask ); if you changed both messages.

```
IMPLEMENT_CO_NETOBJECT_V1(SimpleNetObject)
```

```c
ConsoleMethod(SimpleNetObject, setMessage1
{
    object->setMessage1(argv[2]);
}

ConsoleMethod(SimpleNetObject, setMessage2
{
    object->setMessage2(argv[2]);
}
```

Finally, we use the NetObject implementation macro, IMPLEMENT_CO_NETOBJECT_V1(), to implement our NetObject. It is important that we use this, as it makes Torque perform certain initialization tasks that allow us to send the object over the network. IMPLEMENT_CONOBJECT() doesn't perform these tasks, see the documentation on AbstractClassRep for more details.

See also:
- SimpleNetObject
- NetConnection
SceneObject
Member Function Documentation

void NetObject::clearScopeToClient(NetConnection client)

Undo the effects of a `scopeToClient()` call.

**Parameters:**

*client*  The connection to remove this object's scoping from

**See also:**

`scopeToClient()`

---

int NetObject::getClientObject()

Returns a pointer to the client object when on a local connection.

Short-Circuit-Networking: this is only valid for a local-client / singleplayer situation.

**Returns:**

the `SimObject` ID of the client object.

**Example:**

```cpp
// Psuedo-code, some values left out for 1
%node = new ParticleEmitterNode();
%clientObject = %node.getClientObject();
if(isObject(%clientObject))
    %clientObject.setTransform("0 0 0");
```

**See also:**

Local Connections

---

int NetObject::getGhostID()
Get the ghost index of this object from the server.

**Returns:**

The ghost ID of this NetObject on the server

**Example:**

```plaintext
%ghostID = LocalClientConnection.getGhostId(%serverObject);
```

```plaintext
int NetObject::getServerObject()
```

Returns a pointer to the client object when on a local connection.

Short-Circuit-Netorking: this is only valid for a local-client / singleplayer situation.

**Returns:**

The SimObject ID of the server object.

**Example:**

```plaintext
// Pseudo-code, some values left out for tidiness
%node = new ParticleEmitterNode();
%serverObject = %node.getServerObject();
if(isObject(%serverObject))
    %serverObject.setTransform("0 0 0");
```

**See also:**

Local Connections

```plaintext
bool NetObject::isClientObject()
```

Called to check if an object resides on the client side.
**Returns:**

True if the object resides on the client, false otherwise.

```cpp
bool NetObject::isServerObject()
```

Checks if an object resides on the server.

**Returns:**

True if the object resides on the server, false otherwise.

```cpp
void NetObject::scopeToClient(NetConnection client)
```

Cause the `NetObject` to be forced as scoped on the specified `NetConnection`.

**Parameters:**

- `client` The connection this object will always be scoped to

**Example:**

```cpp
// Called to create new cameras in Torque
// %this - The active GameConnection
// %spawnPoint - The spawn point location
function GameConnection::spawnCamera(%this)
{
    // If this connection's camera exists
    if(isObject(%this.camera))
    {
        // Add it to the mission group to be cleaned up later
        MissionCleanup.add(%this.camera);

        // Force it to scope to the client
        %this.camera.scopeToClient(%this);
    }
}
```
void NetObject::setScopeAlways()

Always scope this object on all connections.

The object is marked as ScopeAlways and is immediately ghosted to all active connections. This function has no effect if the object is not marked as Ghostable.
OcclusionVolume Class Reference
[Miscellaneous]

An invisible shape that causes objects hidden from view behind it to not be rendered. More...

Inheritance diagram for OcclusionVolume:

List of all members.
Public Attributes

**Internal**

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>edge</td>
<td>For internal use only.</td>
</tr>
<tr>
<td>string</td>
<td>plane</td>
<td>For internal use only.</td>
</tr>
<tr>
<td>string</td>
<td>point</td>
<td>For internal use only.</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>Static bool</th>
<th>isRenderable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Static bool</th>
<th>isSelectable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

An invisible shape that causes objects hidden from view behind it to not be rendered.

OcclusionVolume is a class for scene optimization. It's main use is for outdoor spaces where zones and portals do not help in optimizing scene culling as they almost only make sense for modeling visibility in indoor scenarios (and for connecting indoor spaces to outdoor spaces).

During rendering, every object that is fully behind an occluder

Be aware that occluders add overhead to scene culling. Only if this overhead is outweighed by the time saved by not rendering hidden objects, is the occluder actually effective. Because of this, chose only those spots for placing occluders where a significant number of objects will be culled from points that the player will actually be at during the game.

Like zones and portals, OcclusionVolumes may have a default box shape or a more complex

See also:

- Scene::maxOccludersPerZone
- Scene::occluderMinWidthPercentage
- Scene::occluderMinHeightPercentage
Member Data Documentation

string OcclusionVolume::edge

For internal use only.

string OcclusionVolume::plane

For internal use only.

string OcclusionVolume::point

For internal use only.

Copyright © GarageGames, LLC. All Rights Reserved.
OpenFileDialog Class Reference

[File I/O]

Derived from FileDialog, this class is responsible for opening a file browser with the intention of opening a file. More...

Inheritance diagram for OpenFileDialog:

- SimObject
- FileDialog
- OpenFileDialog
- OpenFileDialog

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><strong>MultipleFiles</strong></td>
<td>True/False whether multiple files may be selected and returned or not.</td>
</tr>
<tr>
<td>bool</td>
<td><strong>MustExist</strong></td>
<td>True/False whether the file returned must exist or not.</td>
</tr>
</tbody>
</table>
Detailed Description

Derived from `FileDialog`, this class is responsible for opening a file browser with the intention of opening a file.

The core usage of this dialog is to locate a file in the OS and return the path and name. This does not handle the actual file parsing or data manipulation. That functionality is left up to the `FileObject` class.

Example:

```csharp
// Create a dialog dedicated to opening files
%openFileDlg = new OpenFileDialog()
{
    // Look for jpg image files
    // First part is the descriptor|second part is the extension
    Filters = "Jepg Files|*.jpg";
    // Allow browsing through other folder
    ChangePath = true;

    // Only allow opening of one file at a time
    MultipleFiles = false;
};

// Launch the open file dialog
%result = %openFileDlg.Execute();

// Obtain the chosen file name and path
if ( %result )
{
    %selectedFile = %openFileDlg.file;
}
else
{
    %selectedFile = "";
}
```
// Cleanup
%openFileDlg.delete();

Note:
FileDialog and its related classes are only available in a Tools build of Torque.

See also:
FileDialog
SaveFileDialog
FileObject
# Member Data Documentation

<table>
<thead>
<tr>
<th>bool OpenFileDialog::MultipleFiles</th>
</tr>
</thead>
<tbody>
<tr>
<td>True/False whether multiple files may be selected and returned or not.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool OpenFileDialog::MustExist</th>
</tr>
</thead>
<tbody>
<tr>
<td>True/False whether the file returned must exist or not.</td>
</tr>
</tbody>
</table>
OpenFolderDialog Class Reference

[File I/O]

OS level dialog used for browsing folder structures. More...

Inheritance diagram for OpenFolderDialog:

```
SimObject

FileDialog

OpenFileDialog

OpenFolderDialog
```

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>filename</th>
<th>fileMustExist</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>File that must be in selected folder for it to be valid.</td>
</tr>
</tbody>
</table>
**Detailed Description**

OS level dialog used for browsing folder structures.

This is essentially an OpenFileDialog, but only used for returning directory paths, not files.

**Note:**

FileDialog and its related classes are only available in a Tools build of Torque.

**See also:**

OpenFileDialog for more details on functionality.
<table>
<thead>
<tr>
<th>filename</th>
<th>OpenFolderDialog::fileMustExist</th>
</tr>
</thead>
</table>

File that must be in selected folder for it to be valid.

Copyright © GarageGames, LLC. All Rights Reserved.
ParticleData Class Reference
[Special Effects]

Contains information for how specific particles should look and react including particle colors, particle imagemap, acceleration value for individual particles and spin information. More...

Inheritance diagram for ParticleData:

![Inheritance Diagram](image)

List of all members.
Public Member Functions

void reload ()
Reloads this particle.
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>animateTexture</code></td>
<td>If true, allow the particle texture to be an animated sprite.</td>
</tr>
<tr>
<td>string</td>
<td><code>animTexFrames</code></td>
<td>A list of frames and/or frame ranges to use for particle animation if <code>animateTexture</code> is true.</td>
</tr>
<tr>
<td>string</td>
<td><code>animTexName</code></td>
<td>Texture file to use for this particle if <code>animateTexture</code> is true.</td>
</tr>
<tr>
<td>Point2I</td>
<td><code>animTexTiling</code></td>
<td>The number of frames, in rows and columns stored in <code>textureName</code> (when <code>animateTexture</code> is true).</td>
</tr>
<tr>
<td>float</td>
<td><code>constantAcceleration</code></td>
<td>Constant acceleration to apply to this particle.</td>
</tr>
<tr>
<td>float</td>
<td><code>dragCoefficient</code></td>
<td>Particle physics drag amount.</td>
</tr>
<tr>
<td>int</td>
<td><code>framesPerSec</code></td>
<td>If <code>animateTexture</code> is true, this defines the frames per second of the sprite animation.</td>
</tr>
<tr>
<td>float</td>
<td><code>gravityCoefficient</code></td>
<td>Strength of gravity on the particles.</td>
</tr>
<tr>
<td>float</td>
<td><code>inheritedVelFactor</code></td>
<td>Amount of emitter velocity to add to particle initial velocity.</td>
</tr>
<tr>
<td>int</td>
<td><code>lifetimeMS</code></td>
<td>Time in milliseconds before this particle is destroyed.</td>
</tr>
<tr>
<td>int</td>
<td><code>lifetimeVarianceMS</code></td>
<td>Variance in lifetime of particle, from 0 - <code>lifetimeMS</code>.</td>
</tr>
<tr>
<td>Type</td>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>float</td>
<td>sizes [4]</td>
<td>Particle size keyframe values.</td>
</tr>
<tr>
<td>float</td>
<td>spinRandomMax</td>
<td>Maximum allowed spin speed of this particle, between spinRandomMin and 10000.</td>
</tr>
<tr>
<td>float</td>
<td>spinRandomMin</td>
<td>Minimum allowed spin speed of this particle, between -10000 and spinRandomMax.</td>
</tr>
<tr>
<td>float</td>
<td>spinSpeed</td>
<td>Speed at which to spin the particle.</td>
</tr>
<tr>
<td>Point2F</td>
<td>textureCoords [4]</td>
<td>4 element array defining the UV coords into textureName to use for this particle.</td>
</tr>
<tr>
<td>string</td>
<td>textureName</td>
<td>Texture file to use for this particle.</td>
</tr>
<tr>
<td>float</td>
<td>times [4]</td>
<td>Time keys used with the colors and sizes keyframes.</td>
</tr>
<tr>
<td>bool</td>
<td>useInvAlpha</td>
<td>Controls how particles blend with the scene.</td>
</tr>
<tr>
<td>float</td>
<td>windCoefficient</td>
<td>Strength of wind on the particles.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Contains information for how specific particles should look and react including particle colors, particle imagemap, acceleration value for individual particles and spin information.

**Example:**

```plaintext
datablock ParticleData( GLWaterExpSmoke )
{
    textureName = "art/shapes/particles/smoke"
    dragCoefficient = 0.4;
    gravityCoefficient = -0.25;
    inheritedVelFactor = 0.025;
    constantAcceleration = -1.1;
    lifetimeMS = 1250;
    lifetimeVarianceMS = 0;
    useInvAlpha = false;
    spinSpeed = 1;
    spinRandomMin = -200.0;
    spinRandomMax = 200.0;

    colors[0] = "0.1 0.1 1.0 1.0";
    colors[1] = "0.4 0.4 1.0 1.0";
    colors[2] = "0.4 0.4 1.0 0.0";

    sizes[0] = 2.0;
    sizes[1] = 6.0;
    sizes[2] = 2.0;

    times[0] = 0.0;
    times[1] = 0.5;
    times[2] = 1.0;
};
```
See also:

- ParticleEmitter
- ParticleEmitterData
- ParticleEmitterNode
Member Function Documentation

void ParticleData::reload()

Reloads this particle.

Example:

```
// Get the editor's current particle
%particle = PE_ParticleEditor.currParticle

// Change a particle value
%particle.setFieldVal( %propertyField, %value );

// Reload it
%particle.reload();
```
Member Data Documentation

**bool ParticleData::animateTexture**

If true, allow the particle texture to be an animated sprite.

**string ParticleData::animTexFrames**

A list of frames and/or frame ranges to use for particle animation if animateTexture is true.

Each frame token must be separated by whitespace. A frame token must be a positive integer frame number or a range of frame numbers separated with a '-' . The range separator, '-', cannot have any whitespace around it.

Ranges can be specified to move through the frames in reverse as well as forward (eg. 19-14). Frame numbers exceeding the number of tiles will wrap.

**Example:**

```
animTexFrames = "0-16 20 19 18 17 31-21";
```

**string ParticleData::animTexName**

Texture file to use for this particle if animateTexture is true.

Deprecated. Use textureName instead.

**Point2I ParticleData::animTexTiling**

The number of frames, in rows and columns stored in textureName (when animateTexture is true).
A maximum of 256 frames can be stored in a single texture when using animTexTiling. Value should be "NumColumns NumRows", for example "4 4".

<table>
<thead>
<tr>
<th><strong>ColorF</strong> ParticleData::colors[4]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Particle RGBA color keyframe values.</td>
</tr>
<tr>
<td>The particle color will linearly interpolate between the color/time keys over the lifetime of the particle.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong> ParticleData::constantAcceleration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant acceleration to apply to this particle.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong> ParticleData::dragCoefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>Particle physics drag amount.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>int</strong> ParticleData::framesPerSec</th>
</tr>
</thead>
<tbody>
<tr>
<td>If animateTexture is true, this defines the frames per second of the sprite animation.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong> ParticleData::gravityCoefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strength of gravity on the particles.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong> ParticleData::inheritedVelFactor</th>
</tr>
</thead>
</table>
Amount of emitter velocity to add to particle initial velocity.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int ParticleData::lifetimeMS</td>
<td>Time in milliseconds before this particle is destroyed.</td>
</tr>
<tr>
<td>int ParticleData::lifetimeVarianceMS</td>
<td>Variance in lifetime of particle, from 0 - lifetimeMS.</td>
</tr>
<tr>
<td>float ParticleData::sizes[4]</td>
<td>Particle size keyframe values. The particle size will linearly interpolate between the size/time keys over the lifetime of the particle.</td>
</tr>
<tr>
<td>float ParticleData::spinRandomMax</td>
<td>Maximum allowed spin speed of this particle, between spinRandomMin and 10000.</td>
</tr>
<tr>
<td>float ParticleData::spinRandomMin</td>
<td>Minimum allowed spin speed of this particle, between -10000 and spinRandomMax.</td>
</tr>
<tr>
<td>float ParticleData::spinSpeed</td>
<td></td>
</tr>
</tbody>
</table>
Speed at which to spin the particle.

**Point2F ParticleData::textureCoords[4]**

4 element array defining the UV coords into textureName to use for this particle.

Coords should be set for the first tile only when using animTexTiling; coordinates for other tiles will be calculated automatically. "0 0" is top left and "1 1" is bottom right.

**string ParticleData::textureName**

Texture file to use for this particle.

**float ParticleData::times[4]**

Time keys used with the colors and sizes keyframes.

Values are from 0.0 (particle creation) to 1.0 (end of lifespace).

**bool ParticleData::useInvAlpha**

Controls how particles blend with the scene.

If true, particles blend like ParticleBlendStyle NORMAL, if false, blend like ParticleBlendStyle ADDITIVE.

**Note:**

If ParticleEmitterData::blendStyle is set, it will override this value.
float ParticleData::windCoefficient

Strength of wind on the particles.
ParticleEmitter Class Reference
[Special Effects]

This object is responsible for spawning particles. More...

Inheritance diagram for ParticleEmitter:

List of all members.
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

This object is responsible for spawning particles.

Note:

This class is not normally instantiated directly - to place a simple particle emitting object in the scene, use a ParticleEmitterNode instead.

This class is the main interface for creating particles - though it is usually only accessed from within another object like ParticleEmitterNode or WheeledVehicle. If using this object class (via C++) directly, be aware that it does not track changes in source axis or velocity over the course of a single update, so emitParticles should be called at a fairly fine grain. The emitter will potentially track the last particle to be created into the next call to this function in order to create a uniformly random time distribution of the particles.

If the object to which the emitter is attached is in motion, it should try to ensure that for call (n+1) to this function, start is equal to the end from call (n). This will ensure a uniform spatial distribution.

See also:

ParticleEmitterData
ParticleEmitterNode

Copyright © GarageGames, LLC. All Rights Reserved.
ParticleEmitterData Class Reference
[Special Effects]

Defines particle emission properties such as ejection angle, period and velocity for a ParticleEmitter. More...

Inheritance diagram for ParticleEmitterData:

SimObject

SimDataBlock

GameBaseData

ParticleEmitterData

[legend]

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>void reload ()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reloads the <a href="#">ParticleData</a> datablocks and other fields used by this emitter.</td>
</tr>
</tbody>
</table>
## Public Attributes

### ParticleEmitterData

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point3F</td>
<td><code>alignDirection</code></td>
<td>The direction aligned particles should face, only valid if <code>alignParticles</code> is true.</td>
</tr>
<tr>
<td>bool</td>
<td><code>alignParticles</code></td>
<td>If true, particles always face along the axis defined by <code>alignDirection</code>.</td>
</tr>
<tr>
<td>float</td>
<td><code>ambientFactor</code></td>
<td>Used to generate the final particle color by controlling interpolation between the particle color and the particle color multiplied by the ambient light color.</td>
</tr>
<tr>
<td>ParticleBlendStyle</td>
<td><code>blendStyle</code></td>
<td>String value that controls how emitted particles blend with the scene.</td>
</tr>
<tr>
<td>float</td>
<td><code>ejectionOffset</code></td>
<td>Distance along ejection Z axis from which to eject particles.</td>
</tr>
<tr>
<td>int</td>
<td><code>ejectionPeriodMS</code></td>
<td>Time (in milliseconds) between each particle ejection.</td>
</tr>
<tr>
<td>float</td>
<td><code>ejectionVelocity</code></td>
<td>Particle ejection velocity.</td>
</tr>
<tr>
<td>bool</td>
<td><code>highResOnly</code></td>
<td>This particle system should not use the mixed-resolution renderer. If your particle system has large amounts of overdraw, consider disabling this option.</td>
</tr>
<tr>
<td>int</td>
<td><code>lifetimeMS</code></td>
<td></td>
</tr>
<tr>
<td><strong>Lifetime of emitted particles (in milliseconds).</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| **int** lifetimeVarianceMS  
Variance in particle lifetime from 0 - lifetimeMS. |
| **bool** orientOnVelocity  
If true, particles will be oriented to face in the direction they are moving. |
| **bool** orientParticles  
If true, Particles will always face the camera. |
| **bool** overrideAdvance  
If false, particles emitted in the same frame have their positions adjusted. If true, adjustment is skipped and particles will clump together. |
| **string** particles  
List of space or TAB delimited ParticleData datablock names. |
| **int** periodVarianceMS  
Variance in ejection period, from 1 - ejectionPeriodMS. |
| **float** phiReferenceVel  
Reference angle, from the vertical plane, to eject particles from. |
| **float** phiVariance  
Variance from the reference angle, from 0 - 360. |
| **bool** renderReflection  
Controls whether particles are rendered onto reflective surfaces like water. |
| **bool** reverseOrder  
If true, reverses the normal draw order of particles. |
| **float** softnessDistance |
For soft particles, the distance (in meters) where particles will be faded based on the difference in depth between the particle and the scene geometry.

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>sortParticles</code></td>
<td>If true, particles are sorted furthest to nearest.</td>
</tr>
<tr>
<td>string</td>
<td><code>textureName</code></td>
<td>Optional texture to override <code>ParticleData::textureName</code>.</td>
</tr>
<tr>
<td>float</td>
<td><code>thetaMax</code></td>
<td>Maximum angle, from the horizontal plane, to eject particles from.</td>
</tr>
<tr>
<td>float</td>
<td><code>thetaMin</code></td>
<td>Minimum angle, from the horizontal plane, to eject from.</td>
</tr>
<tr>
<td>bool</td>
<td><code>useEmitterColors</code></td>
<td>If true, use emitter specified colors instead of datablock colors.</td>
</tr>
<tr>
<td>bool</td>
<td><code>useEmitterSizes</code></td>
<td>If true, use emitter specified sizes instead of datablock sizes.</td>
</tr>
<tr>
<td>float</td>
<td><code>velocityVariance</code></td>
<td>Variance for ejection velocity, from 0 - ejectionVelocity.</td>
</tr>
</tbody>
</table>
Detailed Description

Defines particle emission properties such as ejection angle, period and velocity for a ParticleEmitter.

Example:

datablock ParticleEmitterData( GrenadeExpDustEmitter ){
  ejectionPeriodMS = 1;
  periodVarianceMS = 0;
  ejectionVelocity = 15;
  velocityVariance = 0.0;
  ejectionOffset = 0.0;
  thetaMin = 85;
  thetaMax = 85;
  phiReferenceVel = 0;
  phiVariance = 360;
  overrideAdvance = false;
  lifetimeMS = 200;
  particles = "GrenadeExpDust";
};

See also:

ParticleEmitter
ParticleData
ParticleEmitterNode
Member Function Documentation

```cpp
void ParticleEmitterData::reload()
```

Reloads the `ParticleData` datablocks and other fields used by this emitter.

Example:

```cpp
// Get the editor's current particle emitter
%emitter = PE_EmitterEditor.currEmitter

// Change a field value
%emitter.setFieldValue( %propertyField, %\n
// Reload this emitter
%emitter.reload();
```
### Member Data Documentation

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alignDirection</td>
<td>Point3F</td>
<td>The direction aligned particles should face, only valid if alignParticles is true.</td>
</tr>
<tr>
<td>alignParticles</td>
<td>bool</td>
<td>If true, particles always face along the axis defined by alignDirection.</td>
</tr>
<tr>
<td>ambientFactor</td>
<td>float</td>
<td>Used to generate the final particle color by controlling interpolation between the particle color and the particle color multiplied by the ambient light color.</td>
</tr>
<tr>
<td>blendStyle</td>
<td>ParticleBlendStyle</td>
<td>String value that controls how emitted particles blend with the scene.</td>
</tr>
<tr>
<td>ejectionOffset</td>
<td>float</td>
<td>Distance along ejection Z axis from which to eject particles.</td>
</tr>
<tr>
<td>ejectionPeriodMS</td>
<td>int</td>
<td>Time (in milliseconds) between each particle ejection.</td>
</tr>
</tbody>
</table>
```plaintext
float ParticleEmitterData::ejectionVelocity

Particle ejection velocity.

bool ParticleEmitterData::highResOnly

This particle system should not use the mixed-resolution renderer. If your particle system has large amounts of overdraw, consider disabling this option.

int ParticleEmitterData::lifetimeMS

Lifetime of emitted particles (in milliseconds).

int ParticleEmitterData::lifetimeVarianceMS

Variance in particle lifetime from 0 - lifetimeMS.

bool ParticleEmitterData::orientOnVelocity

If true, particles will be oriented to face in the direction they are moving.

bool ParticleEmitterData::orientParticles

If true, Particles will always face the camera.

bool ParticleEmitterData::overrideAdvance
```
If false, particles emitted in the same frame have their positions adjusted. If true, adjustment is skipped and particles will clump together.

**string ParticleEmitterData::particles**

List of space or TAB delimited `ParticleData` datablock names. A random one of these datablocks is selected each time a particle is emitted.

**int ParticleEmitterData::periodVarianceMS**

Variance in ejection period, from 1 - `ejectionPeriodMS`.

**float ParticleEmitterData::phiReferenceVel**

Reference angle, from the vertical plane, to eject particles from.

**float ParticleEmitterData::phiVariance**

Variance from the reference angle, from 0 - 360.

**bool ParticleEmitterData::renderReflection**

Controls whether particles are rendered onto reflective surfaces like water.

**bool ParticleEmitterData::reverseOrder**
If true, reverses the normal draw order of particles.

Particles are normally drawn from newest to oldest, or in Z order (furthest first) if sortParticles is true. Setting this field to true will reverse that order: oldest first, or nearest first if sortParticles is true.

**float ParticleEmitterData::softnessDistance**

For soft particles, the distance (in meters) where particles will be faded based on the difference in depth between the particle and the scene geometry.

**bool ParticleEmitterData::sortParticles**

If true, particles are sorted furthest to nearest.

**string ParticleEmitterData::textureName**

Optional texture to override `ParticleData::textureName`.

**float ParticleEmitterData::thetaMax**

Maximum angle, from the horizontal plane, to eject particles from.

**float ParticleEmitterData::thetaMin**

Minimum angle, from the horizontal plane, to eject from.
bool ParticleEmitterData::useEmitterColors

If true, use emitter specified colors instead of datablock colors.

Useful for ShapeBase dust and WheeledVehicle wheel particle emitters that use the current material to control particle color.

bool ParticleEmitterData::useEmitterSizes

If true, use emitter specified sizes instead of datablock sizes.

Useful for Debris particle emitters that control the particle size.

float ParticleEmitterData::velocityVariance

Variance for ejection velocity, from 0 - ejectionVelocity.

Copyright © GarageGames, LLC. All Rights Reserved.
ParticleEmitterNode Class Reference
[Special Effects]

A particle emitter object that can be positioned in the world and dynamically enabled or disabled. More...

Inheritance diagram for ParticleEmitterNode:

List of all members.
Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void setActive(bool active)</td>
<td>Turns the emitter on or off.</td>
</tr>
<tr>
<td>void setEmitterDataBlock(ParticleEmitterData emitterDatablock=0)</td>
<td>Assigns the datablock for this emitter node.</td>
</tr>
</tbody>
</table>
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>active</td>
<td>Controls whether particles are emitted from this node.</td>
</tr>
<tr>
<td>ParticleEmitterData</td>
<td>emitter</td>
<td>Datablock to use when emitting particles.</td>
</tr>
<tr>
<td>float</td>
<td>velocity</td>
<td>Velocity to use when emitting particles (in the direction of the ParticleEmitterNode object's up (Z) axis).</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td><strong>isRenderable</strong></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td><strong>isSelectable</strong></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

A particle emitter object that can be positioned in the world and dynamically enabled or disabled.

Example:

```cpp
datablock ParticleEmitterNodeData(SimpleEmitterNodeData)
{
    timeMultiple = 1.0;
};

%emitter = new ParticleEmitterNode()
{
    datablock = SimpleEmitterNodeData;
    active = true;
    emitter = FireEmitterData;
    velocity = 3.5;
};

// Dynamically change emitter datablock
%emitter.setEmitterDataBlock(DustEmitterData);
```

Note:
To change the emitter field dynamically (after the ParticleEmitterNode object has been created) you must use the setEmitterDataBlock() method or the change will not be replicated to other clients in the game. Similarly, use the setActive() method instead of changing the active field directly. When changing velocity, you need to toggle setActive() on and off to force the state change to be transmitted to other clients.

See also:
ParticleEmitterNodeData
ParticleEmitterData
Member Function Documentation

void ParticleEmitterNode::setActive (bool active )

Turns the emitter on or off.

**Parameters:**

  *active* New emitter state

void ParticleEmitterNode::setEmitterDataBlock(ParticleEmitterData emitterDatablock)

Assigns the datablock for this emitter node.

**Parameters:**

  *emitterDatablock* ParticleEmitterData datablock to assign

**Example:**

```cpp
// Assign a new emitter datablock
%emitter.setEmitterDatablock( %emitterData
```
**Member Data Documentation**

```plaintext
bool ParticleEmitterNode::active

Controls whether particles are emitted from this node.
```

```plaintext
ParticleEmitterData ParticleEmitterNode::emitter

Datablock to use when emitting particles.
```

```plaintext
float ParticleEmitterNode::velocity

Velocity to use when emitting particles (in the direction of the ParticleEmitterNode object's up (Z) axis).
```
ParticleEmitterNodeData Class Reference

[Special Effects]

Contains additional data to be associated with a ParticleEmitterNode. More...

Inheritance diagram for ParticleEmitterNodeData:

```
SimObject
   ^
  / \
SimDataBlock
   ^  \
  /    \
GameBaseData
   ^  \
  /    \
ParticleEmitterNodeData
```

List of all members.
### Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>timeMultiple</td>
<td>Time multiplier for particle emitter nodes.</td>
</tr>
</tbody>
</table>
Detailed Description

Contains additional data to be associated with a ParticleEmitterNode.
Member Data Documentation

float ParticleEmitterNodeData::timeMultiple

Time multiplier for particle emitter nodes.

Increasing timeMultiple is like running the emitter at a faster rate - single-shot emitters will complete in a shorter time, and continuous emitters will generate particles more quickly.

Valid range is 0.01 - 100.
Path Class Reference
[Miscellaneous]

A spline along which various objects can move along. The spline object acts like a container for Marker objects, which make up the joints, or knots, along the path. Paths can be assigned a speed, can be looping or non-looping. Each of a path's markers can be one of three primary movement types: "normal", "Position Only", or "Kink". More...

Inheritance diagram for Path:

```
SimObject
  └── SimSet
    └── SimGroup
      └── Path
        [legend]
```

List of all members.
Public Member Functions

```c
int getPathId ()
    Returns the PathID (not the object ID) of this path.
```
Public Attributes

<table>
<thead>
<tr>
<th>bool</th>
<th>isLooping</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If this is true, the loop is closed, otherwise it is open.</td>
</tr>
</tbody>
</table>
Detailed Description

A spline along which various objects can move along. The spline object acts like a container for Marker objects, which make up the joints, or knots, along the path. Paths can be assigned a speed, can be looping or non-looping. Each of a path's markers can be one of three primary movement types: "normal", "Position Only", or "Kink".

Example:

```cpp
new path()
{
    isLooping = "1";

    new Marker()
    {
        seqNum = "0";
        type = "Normal";
        msToNext = "1000";
        smoothingType = "Spline";
        position = "-0.054708 -35.0612 234.802";
        rotation = "1 0 0 0";
    };
}
```

See also:

Marker
NetConnection::transmitPaths()
NetConnection::clearPaths()
Path
**Member Function Documentation**

```c++
int Path::getPathId()
```

Returns the PathID (not the object ID) of this path.

**Returns:**

PathID (not the object ID) of this path.

**Example:**

```
// Acquire the PathID of this path object.
%pathID = %thisPath.getPathId();
```
Member Data Documentation

**bool Path::isLooping**

If this is true, the loop is closed, otherwise it is open.
PathCamera Class Reference
[Path Camera]

A camera that moves along a path. The camera can then be made to travel along this path forwards or backwards. More...

Inheritance diagram for PathCamera:

```
  SimObject
    NetObject
      SceneObject
        GameBase
          ShapeBase
            PathCamera
```

List of all members.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>popFront()</code></td>
<td>Removes the knot at the front of the camera's path.</td>
</tr>
<tr>
<td><code>pushBack(TransformF transform, float speed=1.0, string type=&quot;Normal&quot;, string path=&quot;Linear&quot;)</code></td>
<td>Adds a new knot to the back of a path camera's path.</td>
</tr>
<tr>
<td><code>pushFront(TransformF transform, float speed=1.0, string type=&quot;Normal&quot;, string path=&quot;Linear&quot;)</code></td>
<td>Adds a new knot to the front of a path camera's path.</td>
</tr>
<tr>
<td><code>reset(float speed=1.0f)</code></td>
<td>Clear the camera's path and set the camera's current transform as the start of the new path.</td>
</tr>
<tr>
<td><code>setPosition(float position=0.0f)</code></td>
<td>Set the current position of the camera along the path.</td>
</tr>
<tr>
<td><code>setState(string newState=&quot;forward&quot;)</code></td>
<td>Set the movement state for this path camera.</td>
</tr>
<tr>
<td><code>setTarget(float position=1.0f)</code></td>
<td>Set the movement target for this camera along its path.</td>
</tr>
</tbody>
</table>

**Callbacks**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>onNode(string node)</code></td>
<td>A script callback that indicates the path camera has arrived at a specific node in its path. Server side only.</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td><strong>isRenderable</strong></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td><strong>isSelectable</strong></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

A camera that moves along a path. The camera can then be made to travel along this path forwards or backwards.

A camera's path is made up of knots, which define a position, rotation and speed for the camera. Traversal from one knot to another may be either linear or using a Catmull-Rom spline. If the knot is part of a spline, then it may be a normal knot or defined as a kink. Kinked knots are a hard transition on the spline rather than a smooth one. A knot may also be defined as a position only. In this case the knot is treated as a normal knot but is ignored when determining how to smoothly rotate the camera while it is travelling along the path (the algorithm moves on to the next knot in the path for determining rotation).

The datablock field for a PathCamera is a previously created PathCameraData, which acts as the interface between the script and the engine for this PathCamera object.

See also:
PathCameraData

Example:

```plaintext
%newPathCamera = new PathCamera()
{
    dataBlock = LoopingCam;
    position = "0 0 300 1 0 0 0";
};
```
Member Function Documentation

void PathCamera::onNode (string node )

A script callback that indicates the path camera has arrived at a specific node in its path. Server side only.

Parameters:

Node Unique ID assigned to this node.

void PathCamera::popFront( )

Removes the knot at the front of the camera's path.

Example:

// Remove the first knot in the camera's path.
%pathCamera.popFront();

void PathCamera::pushBack (TransformF transform, float speed = 1.0, string type = "Normal", string path = "Linear")

Adds a new knot to the back of a path camera's path.

Parameters:

Transform for the new knot. In the form of "x y z ax ay az aa" such as returned by SceneObject::getTransform()

speed Speed setting for this knot.

type Knot type (Normal, Position Only, Kink).
Example:

```cpp
// Transform vector for new knot. (Pos_X F
%transform = "15.0 5.0 5.0 1.4 1.0 0.2 1.0"

// Speed setting for knot.
%speed = "1.0"

// Knot type. (Normal, Position Only, Kink)
%type = "Normal";

// Path Type. (Linear, Spline)
%path = "Linear";

// Inform the path camera to add a new knot
%pathCamera.pushBack(%transform,%speed,%type, %path);
```

```cpp
void PathCamera::pushFront(TransformF transform, 
    float speed = 1.0, 
    string type = "Normal", 
    string path = "Linear"
)
```

Adds a new knot to the front of a path camera's path.

**Parameters:**

- **transform**: Transform for the new knot. In the form of "x y z ax ay az aa" such as returned by `SceneObject::getTransform()`
- **speed**: Speed setting for this knot.
- **type**: Knot type (Normal, Position Only, Kink).
- **path**: Path type (Linear, Spline).
Example:

```cpp
// Transform vector for new knot. (Pos_X,Y,Z,Rot_X,Y,Z,Angle)
%transform = "15.0 5.0 5.0 1.4 1.0 0.2 1.0"

// Speed setting for knot.
%speed = "1.0";

// Knot type. (Normal, Position Only, Kink)
%type = "Normal";

// Path Type. (Linear, Spline)
%path = "Linear";

// Inform the path camera to add a new knot
%pathCamera.pushFront(%transform, %speed,
```

```cpp
void PathCamera::reset(float speed = 1.0f)
```

Clear the camera's path and set the camera's current transform as the start of the new path.

What specifically occurs is a new knot is created from the camera's current transform. Then the current path is cleared and the new knot is pushed onto the path. Any previous target is cleared and the camera's movement state is set to Forward. The camera is now ready for a new path to be defined.

**Parameters:**

- `speed` Speed for the camera to move along its path after being reset.

**Example:**

```
// Determine the new movement speed of this
```
%speed = "0.50";

// Inform the path camera to start a new path at the camera's current position, and set the new path's speed value.
%pathCamera.reset(%speed);

void PathCamera::setPosition(float position = 0.0f)

Set the current position of the camera along the path.

**Parameters:**

*position* Position along the path, from 0.0 (path start) - 1.0 (path end), to place the camera.

**Example:**

```c++
// Set the camera on a position along its path
%position = "0.35";

// Force the pathCamera to its new position
%pathCamera.setPosition(%position);
```

void PathCamera::setState(string newState = "forward")

Set the movement state for this path camera.

**Parameters:**

*newState* New movement state type for this camera. Forward, Backward or Stop.

**Example:**

```c++
// Set the state type (forward, backward, stop)
// In this example, the camera will travel to the last node (or target if given) w:
%state = "forward";

// Inform the pathCamera to change its mo\n%pathCamera.setState(%state);

void PathCamera::setTarget(float position = 1.0f)

Set the movement target for this camera along its path.

The camera will attempt to move along the path to the given target in the direction provided by setState() (the default is forwards). Once the camera moves past this target it will come to a stop, and the target state will be cleared.

**Parameters:**

*position* Target position, between 0.0 (path start) and 1.0 (path end), for the camera to move to along its path.

**Example:**

```
// Set the position target, between 0.0 (path start) and 1.0 (path end), for the camera to move to along its path.
%position = "0.50";

// Inform the pathCamera of the new target
%pathCamera.setTarget(%position);
```
PathCameraData Class Reference

[Path Camera]

General interface to control a PathCamera object from the script level. More...

Inheritance diagram for PathCameraData:

```
SimObject
  ↓
SimDataBlock
  ↓
GameBaseData
  ↓
ShapeBaseData
  ↓
PathCameraData
```

List of all members.
Detailed Description

General interface to control a PathCamera object from the script level.

See also:
PathCamera

Example:

```datablock PathCameraData(LoopingCam)
{
    mode = "";
}
```
PfxVis Class Reference

Singleton class that exposes ConsoleStaticFunctions for debug visualizing PostEffects. More...

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>clear ()</strong></td>
<td>Close all visualization windows.</td>
</tr>
<tr>
<td><strong>hide ()</strong></td>
<td>Hide all visualization windows (they are not destroyed).</td>
</tr>
<tr>
<td>*<em>onWindowClosed (GuiWindowCtrl <em>ctrl)</em></em></td>
<td>Callback when a visualization window is closed.</td>
</tr>
<tr>
<td><strong>open (PostEffect effect, bool clear)</strong></td>
<td>Open visualization windows for all input and target textures.</td>
</tr>
<tr>
<td><strong>show ()</strong></td>
<td>Show all visualization windows.</td>
</tr>
</tbody>
</table>
Detailed Description

Singleton class that exposes ConsoleStaticFunctions for debug visualizing PostEffects.

Example:

```cpp
// Script interface...
PfxVis::open( PostEffect ) // Multiple PostEffects can be visualized at the same time
PfxVis::clear() // Clear all visualizer windows
PfxVis::hide() // Hide all windows (not destroyed)
PfxVis::show()
```
Member Function Documentation

void PfxVis::clear()  
Close all visualization windows.

Example:

    PfxVis::clear();

void PfxVis::hide()  
Hide all visualization windows (they are not destroyed).

Example:

    PfxVis::hide();

void PfxVis::onWindowClosed(GuiWindowCtrl * ctrl)  
Callback when a visualization window is closed.

Parameters:

    ctrl  Name of the GUI control being closed

Example:

    PfxVis::onWindowClosed( VisWindow )

void PfxVis::open(PostEffect effect, bool clear)
Open visualization windows for all input and target textures.

**Parameters:**

- `effect` Name of the PostEffect to open
- `clear` True to close all visualization windows before opening the effect

**Example:**

```cpp
// Multiple PostEffects can be visualized
PfxVis::open( PostEffect )
```

```cpp
void PfxVis::show( )
```

Show all visualization windows.

**Example:**

```cpp
PfxVis::show();
```
PhysicalZone Class Reference
[Miscellaneous]

Physical Zones are areas that modify the player's gravity and/or velocity and/or applied force. More...

Inheritance diagram for PhysicalZone:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void activate()</code></td>
<td>Activate the physical zone's effects.</td>
</tr>
<tr>
<td><code>void deactivate()</code></td>
<td>Deactivate the physical zone's effects.</td>
</tr>
</tbody>
</table>
### Public Attributes

#### Misc

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point3F</td>
<td><code>appliedForce</code></td>
<td>Three-element floating point value representing forces in three axes to apply to objects entering <strong>PhysicalZone</strong>.</td>
</tr>
<tr>
<td>float</td>
<td><code>gravityMod</code></td>
<td>Gravity in <strong>PhysicalZone</strong>. Multiplies against standard gravity.</td>
</tr>
<tr>
<td>floatList</td>
<td><code>polyhedron</code></td>
<td>The polyhedron type is really a quadrilateral and consists of a cornerpoint followed by three vectors representing the edges extending from the corner.</td>
</tr>
<tr>
<td>float</td>
<td><code>velocityMod</code></td>
<td>Multiply velocity of objects entering zone by this value every tick.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>static bool</th>
<th><strong>isRenderable</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td><strong>isSelectable</strong></td>
</tr>
<tr>
<td></td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td><strong>renderZones</strong></td>
</tr>
<tr>
<td></td>
<td>If true, a box will render around the location of all PhysicalZones.</td>
</tr>
</tbody>
</table>
Detailed Description

Physical Zones are areas that modify the player's gravity and/or velocity and/or applied force.

The datablock properties determine how the physics, velocity and applied forces affect a player who enters this zone.

Example:

```cpp
new PhysicalZone(Team1JumpPad) {
  velocityMod = "1"; gravityMod = "0";
  appliedForce = "0 0 20000";
  polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000 1.0000000";
  position = "273.559 -166.371 249.856";
  rotation = "0 0 1 13.0216";
  scale = "8 4.95 28.31";
  isRenderEnabled = "true";
  canSaveDynamicFields = "1";
  enabled = "1";
};
```
Member Function Documentation

void PhysicalZone::activate()

Activate the physical zone's effects.

Example:

// Activate effects for a specific physical zone.
%thisPhysicalZone.activate();

void PhysicalZone::deactivate()

Deactivate the physical zone's effects.

Example:

// Deactivate effects for a specific physical zone.
%thisPhysicalZone.deactivate();
# Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point3F</td>
<td>PhysicalZone::appliedForce</td>
<td>Three-element floating point value representing forces in three axes to apply to objects entering PhysicalZone.</td>
</tr>
<tr>
<td>float</td>
<td>PhysicalZone::gravityMod</td>
<td>Gravity in PhysicalZone. Multiplies against standard gravity.</td>
</tr>
<tr>
<td>floatList</td>
<td>PhysicalZone::polyhedron</td>
<td>The polyhedron type is really a quadrilateral and consists of a cornerpoint followed by three vectors representing the edges extending from the corner.</td>
</tr>
<tr>
<td>bool</td>
<td>PhysicalZone::renderZones [static]</td>
<td>If true, a box will render around the location of all PhysicalZones.</td>
</tr>
<tr>
<td>float</td>
<td>PhysicalZone::velocityMod</td>
<td>Multiply velocity of objects entering zone by this value every tick.</td>
</tr>
</tbody>
</table>
PhysicsDebris Class Reference

[Physics]

Represents one or more rigid bodies defined in a single mesh file with a limited lifetime. **More...**

Inheritance diagram for PhysicsDebris:

```
SimObject
  ↓
NetObject
  ↓
SceneObject
  ↓
GameBase
  ↓
PhysicsDebris
```

[legend]

List of all members.
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Represents one or more rigid bodies defined in a single mesh file with a limited lifetime.

A PhysicsDebris object can be viewed as a single system capable of generating multiple PhysicsBodies as debris when triggered. Vaguely similar to how a ParticleEmitter is capable of creating Particles, but isn't a particle in itself. After it's lifetime has elapsed, the object will be deleted.

PhysicsDebris loads a standard .DAE or .DTS file and creates a rigid body for each defined collision node.

For collision nodes to work correctly, they must be setup as follows:

- Visible mesh nodes are siblings of the collision node under a common parent dummy node.
- Collision node is a child of its visible mesh node.

Colmesh type nodes are NOT supported; physx and most standard rigid body simulations do not support arbitrary triangle meshes for dynamics do to the computational expense.

Therefore, collision nodes must be one of the following:

- Colbox
- Colsphere
- Colcapsule
- Col (convex).

PhysicsDebris should NOT be created on the server.
PhysicsDebrisData Class Reference

[Physics]

Defines the properties of a PhysicsDebris object. More...

Inheritance diagram for PhysicsDebrisData:

```
legend

SimObject

SimDataDblock

GameBaseData

PhysicsDebrisData
```

List of all members.
### Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>preload</td>
<td>Loads some information to have readily available at simulation time.</td>
</tr>
</tbody>
</table>

### Physical Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>angularDamping</td>
<td>Value that reduces an object’s rotational velocity over time.</td>
</tr>
<tr>
<td>float</td>
<td>angularSleepThreshold</td>
<td>Minimum rotational velocity before the shape can be put to sleep.</td>
</tr>
<tr>
<td>float</td>
<td>buoyancyDensity</td>
<td>The density of this shape for purposes of calculating buoyant forces.</td>
</tr>
<tr>
<td>float</td>
<td>friction</td>
<td>Coefficient of kinetic friction to be applied to the shape.</td>
</tr>
<tr>
<td>float</td>
<td>lifetime</td>
<td>Base time, in seconds, that debris persists after time of creation.</td>
</tr>
<tr>
<td>float</td>
<td>lifetimeVariance</td>
<td>Range of variation randomly applied to lifetime when debris is created.</td>
</tr>
<tr>
<td>float</td>
<td>linearDamping</td>
<td>Value that reduces an object's linear velocity over time.</td>
</tr>
<tr>
<td>float</td>
<td>linearSleepThreshold</td>
<td>Minimum linear velocity before the shape can be put to sleep.</td>
</tr>
<tr>
<td>float</td>
<td>mass</td>
<td>Value representing the mass of the shape.</td>
</tr>
<tr>
<td>Type</td>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>float</td>
<td>restitution</td>
<td>Bounce coefficient applied to the shape in response to a collision.</td>
</tr>
<tr>
<td>float</td>
<td>staticFriction</td>
<td>Coefficient of static friction to be applied to the shape.</td>
</tr>
<tr>
<td>float</td>
<td>waterDampingScale</td>
<td>Scale to apply to linear and angular dampening while underwater.</td>
</tr>
</tbody>
</table>

**Display**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>castShadows</td>
<td>Determines if the shape's shadow should be cast onto the environment.</td>
</tr>
<tr>
<td>filename</td>
<td>shapeFile</td>
<td>Path to the .DAE or .DTS file to use for this shape.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Defines the properties of a *PhysicsDebris* object.

**See also:**

  *PhysicsDebris.*
**Member Data Documentation**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>PhysicsDebrisData::angularDamping</td>
<td>Value that reduces an object's rotational velocity over time. Larger values will cause velocity to decay quicker.</td>
</tr>
<tr>
<td>float</td>
<td>PhysicsDebrisData::angularSleepThreshold</td>
<td>Minimum rotational velocity before the shape can be put to sleep. This should be a positive value. Shapes put to sleep will not be simulated in order to save system resources.</td>
</tr>
</tbody>
</table>

**Note:**
The shape must be dynamic.

| float  | PhysicsDebrisData::buoyancyDensity        | The density of this shape for purposes of calculating buoyant forces. The result of the calculated buoyancy is relative to the density of the WaterObject the PhysicsDebris is within. |

**See also:**
WaterObject::density

| bool   | PhysicsDebrisData::castShadows            | Determines if the shape's shadow should be cast onto the environment. |


float PhysicsDebrisData::friction

Coefficient of kinetic friction to be applied to the shape.

Kinetic friction reduces the velocity of a moving object while it is in contact with a surface. A larger coefficient will result in a larger reduction in velocity. A shape's friction should be smaller than it's staticFriction, but greater than 0.

**Note:**
This value is only applied while an object is in motion. For an object starting at rest, see PhysicsDebrisData::staticFriction

float PhysicsDebrisData::lifetime

Base time, in seconds, that debris persists after time of creation.

**Note:**
A PhysicsDebris' lifetime multiplied by it's $pref::PhysicsDebris::lifetimeScale must be equal to or greater than 1.0.

float PhysicsDebrisData::lifetimeVariance

Range of variation randomly applied to lifetime when debris is created.

Represents the maximum amount of seconds that will be added or subtracted to a shape's base lifetime. A value of 0 will apply the same lifetime to each shape created.

float PhysicsDebrisData::linearDamping
Value that reduces an object's linear velocity over time. Larger values will cause velocity to decay quicker.

```markdown
float PhysicsDebrisData::linearSleepThreshold
```

Minimum linear velocity before the shape can be put to sleep. This should be a positive value. Shapes put to sleep will not be simulated in order to save system resources.

**Note:**
The shape must be dynamic.

```markdown
float PhysicsDebrisData::mass
```

Value representing the mass of the shape. A shape's mass influences the magnitude of any force applied to it.

**Note:**
All `PhysicsDebris` objects are dynamic.

```markdown
void PhysicsDebrisData::preload
```

Loads some information to have readily available at simulation time.

Forces generation of shaders, materials, and other data used by the PhysicsDebris object. This function should be used while a level is loading in order to shorten the amount of time to create a `PhysicsDebris` in game.
float PhysicsDebrisData::restitution

Bounce coeffecient applied to the shape in response to a collision.

Restitution is a ratio of a shape's velocity before and after a collision. A value of 0 will zero out a shape's post-collision velocity, making it stop on contact. Larger values will remove less velocity after a collision, making it 'bounce' with greater force. Normal restitution values range between 0 and 1.0.

Note:
Values near or equaling 1.0 are likely to cause undesirable results in the physics simulation. Because of this, it is recommended to avoid values close to 1.0

filename PhysicsDebrisData::shapeFile

Path to the .DAE or .DTS file to use for this shape.

Compatible with Live-Asset Reloading.

float PhysicsDebrisData::staticFriction

Coefficient of static friction to be applied to the shape.

Static friction determines the force needed to start moving an at-rest object in contact with a surface. If the force applied onto shape cannot overcome the force of static friction, the shape will remain at rest. A higher coefficient will require a larger force to start motion. This value should be both greater than 0 and the PhysicsDebrisData::friction.

Note:
This value is only applied while an object is at rest. For an object in motion, see PhysicsDebrisData::friction
float PhysicsDebrisData::waterDampingScale

Scale to apply to linear and angular dampening while underwater.

See also:
  angularDamping linearDamping
PhysicsForce Class Reference

Helper object for gameplay physical forces. More...

Inheritance diagram for PhysicsForce:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>void attach (Point3F start, Point3F direction, float maxDist)</strong></td>
<td>Attempts to associate the <em>PhysicsForce</em> with a <em>PhysicsBody</em>.</td>
</tr>
<tr>
<td><strong>void detach (Point3F force=Point3F::Zero)</strong></td>
<td>Disassociates the <em>PhysicsForce</em> from any attached <em>PhysicsBody</em>.</td>
</tr>
<tr>
<td><strong>bool isAttached ()</strong></td>
<td>Returns true if the <em>PhysicsForce</em> is currently attached to an object.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>static bool</th>
<th><code>isRenderable</code></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static bool</th>
<th><code>isSelectable</code></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Helper object for gameplay physical forces.

PhysicsForces can be created and "attached" to other PhysicsBodies to attract them to the position of the PhysicsForce.
void PhysicsForce::attach(Point3F start,
               Point3F direction,
               float maxDist)

Attempts to associate the PhysicsForce with a PhysicsBody.

Performs a physics ray cast of the provided length and direction. The PhysicsForce will attach itself to the first dynamic PhysicsBody the ray collides with. On every tick, the attached body will be attracted towards the position of the PhysicsForce.

A PhysicsForce can only be attached to one body at a time.

Note:
   To determine if an attach was successful, check isAttached() immediately after calling this function.

void PhysicsForce::detach(Point3F force = Point3F::Zero)

Disassociates the PhysicsForce from any attached PhysicsBody.

Parameters:
   force  Optional force to apply to the attached PhysicsBody before detaching.

Note:
   Has no effect if the PhysicsForce is not attached to anything.

bool PhysicsForce::isAttached()

Returns true if the PhysicsForce is currently attached to an object.
See also:

PhysicsForce::attach()
PhysicsShape Class Reference

[Maths]

Represents a destructible physical object simulated through the plugin system. More...

Inheritance diagram for PhysicsShape:

![Inheritance Diagram]

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void destroy ()</code></td>
<td>Disables rendering and physical simulation.</td>
</tr>
<tr>
<td><code>bool isDestroyed ()</code></td>
<td>Returns if a <code>PhysicsShape</code> has been destroyed or not.</td>
</tr>
<tr>
<td><code>void restore ()</code></td>
<td>Restores the shape to its state before being destroyed.</td>
</tr>
</tbody>
</table>
Public Attributes

PhysicsShape

bool playAmbient

Enables or disables playing of an ambient animation upon loading the shape.
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>noCorrections</td>
<td>Determines if the shape will receive corrections from the server or will instead be allowed to diverge.</td>
</tr>
<tr>
<td>static bool</td>
<td>noSmoothing</td>
<td>Determines if client-side shapes will attempt to smoothly transition to their new position after receiving a correction.</td>
</tr>
</tbody>
</table>
Detailed Description

Represents a destructible physical object simulated through the plugin system.

See also:

PhysicsShapeData.
Member Function Documentation

void PhysicsShape::destroy( )

Disables rendering and physical simulation.

Calling `destroy()` will also spawn any explosions, debris, and/or destroyedShape defined for it, as well as remove it from the scene graph.

Destroyed objects are only created on the server. Ghosting will later update the client.

**Note:**

This does not actually delete the PhysicsShape.

bool PhysicsShape::isDestroyed( )

Returns if a PhysicsShape has been destroyed or not.

void PhysicsShape::restore( )

Restores the shape to its state before being destroyed.

Re-enables rendering and physical simulation on the object and adds it to the client's scene graph. Has no effect if the shape is not destroyed.
**Member Data Documentation**

| bool PhysicsShape::playAmbient |

Enables or disables playing of an ambient animation upon loading the shape.

**Note:**

The ambient animation must be named "ambient".

Copyright © GarageGames, LLC. All Rights Reserved.
PhysicsShapeData Class Reference
[Physics]

Defines the properties of a PhysicsShape. More...

Inheritance diagram for PhysicsShapeData:

```
  SimObject
    SimDataBlock
      GameBaseData
        PhysicsShapeData
```

List of all members.
## Public Attributes

### Physics

<table>
<thead>
<tr>
<th>Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>angularDamping</td>
<td>Value that reduces an object's rotational velocity over time.</td>
</tr>
<tr>
<td>float</td>
<td>angularSleepThreshold</td>
<td>Minimum rotational velocity before the shape can be put to sleep.</td>
</tr>
<tr>
<td>float</td>
<td>buoyancyDensity</td>
<td>The density of the shape for calculating buoyant forces.</td>
</tr>
<tr>
<td>float</td>
<td>friction</td>
<td>Coefficient of kinetic friction to be applied to the shape.</td>
</tr>
<tr>
<td>float</td>
<td>linearDamping</td>
<td>Value that reduces an object's linear velocity over time.</td>
</tr>
<tr>
<td>float</td>
<td>linearSleepThreshold</td>
<td>Minimum linear velocity before the shape can be put to sleep.</td>
</tr>
<tr>
<td>float</td>
<td>mass</td>
<td>Value representing the mass of the shape.</td>
</tr>
<tr>
<td>float</td>
<td>restitution</td>
<td>Coefficient of a bounce applied to the shape in response to a collision.</td>
</tr>
<tr>
<td>float</td>
<td>staticFriction</td>
<td>Coefficient of static friction to be applied to the shape.</td>
</tr>
<tr>
<td>float</td>
<td>waterDampingScale</td>
<td>Scale to apply to linear and angular</td>
</tr>
</tbody>
</table>
dampening while underwater.

**Media**

<table>
<thead>
<tr>
<th><strong>PhysicsDebrisData</strong></th>
<th><strong>Debris</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of a <strong>PhysicsDebrisData</strong> to spawn when this shape is destroyed (optional).</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>PhysicsShapeData</strong></th>
<th><strong>destroyedShape</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of a <strong>PhysicsShapeData</strong> to spawn when this shape is destroyed (optional).</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>ExplosionData</strong></th>
<th><strong>Explosion</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of an <strong>ExplosionData</strong> to spawn when this shape is destroyed (optional).</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>filename</strong></th>
<th><strong>shapeName</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Path</strong> to the .DAE or .DTS file to use for this shape.</td>
<td></td>
</tr>
</tbody>
</table>

**Networking**

<table>
<thead>
<tr>
<th><strong>PhysicsSimType</strong></th>
<th><strong>simType</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Controls whether this shape is simulated on the server, client, or both physics simulations.</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

Defines the properties of a PhysicsShape.

See also:

PhysicsShape.
### Member Data Documentation

**float** PhysicsShapeData::angularDamping

Value that reduces an object's rotational velocity over time. Larger values will cause velocity to decay quicker.

**float** PhysicsShapeData::angularSleepThreshold

Minimum rotational velocity before the shape can be put to sleep. This should be a positive value. Shapes put to sleep will not be simulated in order to save system resources.

**Note:**

The shape must be dynamic.

**float** PhysicsShapeData::buoyancyDensity

The density of the shape for calculating buoyant forces. The result of the calculated buoyancy is relative to the density of the WaterObject the PhysicsShape is within.

**See also:**

WaterObject::density

**PhysicsDebrisData** PhysicsShapeData::Debris

Name of a PhysicsDebrisData to spawn when this shape is destroyed (optional).
**PhysicsShapeData**: `PhysicsShapeData::destroyedShape`

Name of a **PhysicsShapeData** to spawn when this shape is destroyed (optional).

**ExplosionData**: `PhysicsShapeData::Explosion`

Name of an **ExplosionData** to spawn when this shape is destroyed (optional).

**float**: `PhysicsShapeData::friction`

Coefficient of kinetic friction to be applied to the shape.

Kinetic friction reduces the velocity of a moving object while it is in contact with a surface. A higher coefficient will result in a larger velocity reduction. A shape's friction should be lower than its staticFriction, but larger than 0.

**Note:**

This value is only applied while an object is in motion. For an object starting at rest, see **PhysicsShape::staticFriction**

**float**: `PhysicsShapeData::linearDamping`

Value that reduces an object's linear velocity over time.

Larger values will cause velocity to decay quicker.

**float**: `PhysicsShapeData::linearSleepThreshold`

Minimum linear velocity before the shape can be put to sleep.
This should be a positive value. Shapes put to sleep will not be simulated in order to save system resources.

**Note:**

The shape must be dynamic.

```cpp
float PhysicsShapeData::mass
```

Value representing the mass of the shape.

A shape's mass influences the magnitude of any force exerted on it. For example, a `PhysicsShape` with a large mass requires a much larger force to move than the same shape with a smaller mass.

**Note:**

A mass of zero will create a kinematic shape while anything greater will create a dynamic shape.

```cpp
float PhysicsShapeData::restitution
```

Coefficient of a bounce applied to the shape in response to a collision.

Restitution is a ratio of a shape's velocity before and after a collision. A value of 0 will zero out a shape's post-collision velocity, making it stop on contact. Larger values will remove less velocity after a collision, making it 'bounce' with a greater force. Normal restitution values range between 0 and 1.0.

**Note:**

Values near or equaling 1.0 are likely to cause undesirable results in the physics simulation. Because of this it is recommended to avoid values close to 1.0.
**PhysicsShapeData::filename**: 
Path to the .DAE or .DTS file to use for this shape. 
Compatible with Live-Asset Reloading.

**PhysicsShapeData::PhysicsSimType**: 
Controls whether this shape is simulated on the server, client, or both physics simulations.

**PhysicsShapeData::float staticFriction**: 
Coefficient of static friction to be applied to the shape.

Static friction determines the force needed to start moving an at-rest object in contact with a surface. If the force applied onto shape cannot overcome the force of static friction, the shape will remain at rest. A larger coefficient will require a larger force to start motion. This value should be larger than zero and the physicsShape's friction.

**Note:**
This value is only applied while an object is at rest. For an object in motion, see PhysicsShape::friction

**PhysicsShapeData::waterDampingScale**: 
Scale to apply to linear and angular dampening while underwater. 
Used with the waterViscosity of the

**See also:**
angularDamping linearDamping
Player Class Reference
[Game Objects]

A client-controlled player character. More...

Inheritance diagram for Player:

List of all members.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void allowAllPoses()</code></td>
<td>Allow all poses a chance to occur.</td>
</tr>
<tr>
<td><code>void allowCrouching(bool state)</code></td>
<td>Set if the Player is allowed to crouch.</td>
</tr>
<tr>
<td><code>void allowJetJumping(bool state)</code></td>
<td>Set if the Player is allowed to jet jump.</td>
</tr>
<tr>
<td><code>void allowJumping(bool state)</code></td>
<td>Set if the Player is allowed to jump.</td>
</tr>
<tr>
<td><code>void allowProne(bool state)</code></td>
<td>Set if the Player is allowed to go prone.</td>
</tr>
<tr>
<td><code>void allowSprinting(bool state)</code></td>
<td>Set if the Player is allowed to sprint.</td>
</tr>
<tr>
<td><code>void allowSwimming(bool state)</code></td>
<td>Set if the Player is allowed to swim.</td>
</tr>
<tr>
<td><code>bool checkDismountPoint(Point3F oldPos, Point3F pos)</code></td>
<td>Check if it is safe to dismount at this position.</td>
</tr>
<tr>
<td><code>void clearControlObject()</code></td>
<td>Clears the player's current control object.</td>
</tr>
<tr>
<td><code>int getControlObject()</code></td>
<td>Get the current object we are controlling.</td>
</tr>
<tr>
<td><code>string getDamageLocation(Point3F pos)</code></td>
<td>Get the named damage location and modifier for a given world position.</td>
</tr>
<tr>
<td><code>int getNumDeathAnimations()</code></td>
<td>Get the number of death animations available to this player.</td>
</tr>
<tr>
<td><code>string getPose()</code></td>
<td>Get the name of the player's current pose.</td>
</tr>
<tr>
<td><code>string getState()</code></td>
<td></td>
</tr>
</tbody>
</table>
Get the name of the player's current state.

| bool setActionThread (string name, bool hold=false, bool fsp=true) |
| Set the main action sequence to play for this player. |

| bool setArmThread (string name) |
| Set the sequence that controls the player's arms (dynamically adjusted to match look direction). |

| bool setControlObject (ShapeBase obj) |
| Set the object to be controlled by this player. |
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static int</td>
<td>crouchTrigger</td>
<td>The move trigger index used for player crouching.</td>
</tr>
<tr>
<td>static int</td>
<td>imageTrigger0</td>
<td>The move trigger index used to trigger mounted image 0.</td>
</tr>
<tr>
<td>static int</td>
<td>imageTrigger1</td>
<td>The move trigger index used to trigger mounted image 1 or alternate fire on mounted image 0.</td>
</tr>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>static int</td>
<td>jumpJetTrigger</td>
<td>The move trigger index used for player jump jetting.</td>
</tr>
<tr>
<td>static int</td>
<td>jumpTrigger</td>
<td>The move trigger index used for player jumping.</td>
</tr>
<tr>
<td>static float</td>
<td>maxImpulseVelocity</td>
<td>The maximum velocity allowed due to a single impulse.</td>
</tr>
<tr>
<td>static int</td>
<td>maxPredictionTicks</td>
<td>Maximum number of ticks to predict on the client from the last known move obtained from the server.</td>
</tr>
<tr>
<td>static int</td>
<td>maxWarpTicks</td>
<td>When a warp needs to occur due to the client being too far off from the server, this is the maximum number of ticks we'll allow the client to warp to catch up.</td>
</tr>
<tr>
<td>static float</td>
<td>minWarpTicks</td>
<td>Fraction of tick at which instant warp occurs on the client.</td>
</tr>
<tr>
<td>static int</td>
<td>proneTrigger</td>
<td></td>
</tr>
<tr>
<td>static bool</td>
<td>renderCollision</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Determines if the player's collision mesh should be rendered.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static bool</th>
<th>renderMyItems</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Determines if mounted shapes are rendered or not.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static bool</th>
<th>renderMyPlayer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Determines if the player is rendered or not.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static int</th>
<th>sprintTrigger</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The move trigger index used for player sprinting.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static int</th>
<th>vehicleDismountTrigger</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The move trigger index used to dismount player.</td>
</tr>
</tbody>
</table>
Detailed Description

A client-controlled player character.

The Player object is the main client-controlled object in an FPS, or indeed, any game where the user is in control of a single character. This class (and the associated datablock, PlayerData) allows you to fine-tune the movement, collision detection, animation, and SFX properties of the character. Player derives from ShapeBase, so it is recommended to have a good understanding of that class (and it's parent classes) as well.

See also:
- SceneObject
- GameBase
- ShapeBase
- ShapeBaseImageData
Movement

The Player class supports the following modes of movement, known as poses:

- Stand
- Sprinting
- Crouching
- Prone
- Swimming

The acceleration, maximum speed, and bounding box for each mode can be set independently using the PlayerData datablock. The player will automatically switch between swimming and one of the other 4 'dry' modes when entering/exiting the water, but transitions between the non-swimming modes are handled by controller input (such as holding down a key to begin crouching). $mvTriggerCount3 activates crouching, while $mvTriggerCount4 activates being prone.

It is important to set the bounding box correctly for each mode so that collisions with the player remain accurate:

When the player changes pose a new PlayerData callback onPoseChange() is called. This is being used as
Armor::onPoseChange() to modify an animation prefix used by ShapeBaseImageData to allow the 1st person arms to change their animation based on pose.

Example:

```lisp
function Armor::onPoseChange(%this, %obj, %oldPose, %newPose)
{
    // Set the script anim prefix to be that of the current pose
    %obj.setImageScriptAnimPrefix($WeaponSlot);
}
```

Another feature is being able to lock out poses for the Player at any time. This is done with allowCrouch(), allowSprinting() etc. (there is even allowJumping() and allowJetJumping() which aren't actually poses but states). So if for some game play reason the player should not be allowed to crouch right now, that may be disabled. All poses may be allowed with allowAllPoses() on the Player class.

The pose lock out mechanism is being used by the weapon script system -- see Weapon::onUse(). With this system, weapons can prevent the player from going into certain poses. This is used by the deployable turret to lock out sprinting while the turret is the current weapon.

Example:

```lisp
function Weapon::onUse(%data, %obj)
{
    // Default behavior for all weapons is to:
    // slot, which is currently assumed to be slot 0
    if (%obj.getMountedImage($WeaponSlot) != %data.image.getId()) {
        serverPlay3D(WeaponUseSound, %obj.getTransform());
        %obj.mountImage(%data.image, $WeaponSlot);
    }
    if (%obj.client) {
        //...
    }
}
```
if (%data.description != "")
    messageClient(%obj.client, 'MsgWeaponUsed');
else
    messageClient(%obj.client, 'MsgWeaponUsed');
}

// If this is a Player class object
if (%obj.isInNamespaceHierarchy("Player"))
{
    // Start by allowing everything
    %obj.allowAllPoses();

    // Now see what isn't allowed by
    %image = %data.image;

    if (%image.jumpingDisallowed)
        %obj.allowJumping(false);

    if (%image.jetJumpingDisallowed)
        %obj.allowJetJumping(false);

    if (%image.sprintDisallowed)
        %obj.allowSprinting(false);

    if (%image.crouchDisallowed)
        %obj.allowCrouching(false);

    if (%image.proneDisallowed)
        %obj.allowProne(false);

    if (%image.swimmingDisallowed)
%obj.allowSwimming(false);
}
Sprinting

As mentioned above, sprinting is another pose for the Player class. It defines its own force and max speed in the three directions in PlayerData just like most poses, such as crouch. It is activated using $mvTriggerCount5 by default which is often connected to Left Shift. When used this way you could treat it just like a standard run -- perhaps with the standard pose used for a walk in a RPG.

But sprinting is special in that you can control if a player's movement while sprinting should be constrained. You can place scale factors on strafing, yaw and pitch. These force the player to move mostly in a straight line (or completely if you set them to 0) while sprinting by limiting their motion. You can also choose if the player can jump while sprinting. This is all set up in PlayerData.

Just like other poses, you can define which sequences should be played on the player while sprinting. These sequences are:

- `sprint_root`
- `sprint_forward`
- `sprint_backward`
- `sprint_side`
- `sprint_right`

However, if any of these sequences are not defined for the player, then the standard root, run, back, side and side_right sequences will be used. The idea here is that the ground transform for these sequences will force them to play faster to give the appearance of sprinting. But if you want the player to do something different than just look like they're running faster -- such as holding their weapon against their body -- then you'll want to make use of the sprint specific sequences.

Sprint also provides two PlayerData callbacks: `onStartSprintMotion()` and `onStopSprintMotion()`. The start callback is called when the player is in a sprint pose and starts to move (i.e. presses the W key). The stop callback is called when either the player stops moving, or they stop sprinting. These could be used for anything, but by default
they are tied into the `ShapeBaseImageData` system. See `Armor::onStartSprintMotion()` and `Armor::onStopSprintMotion()`. With `ShapeBaseImageData` supporting four generic triggers that may be used by a weapon's state machine to do something, the first one is triggered to allow weapons to enter a special sprint state that plays a sprint animation sequence and locks out firing. However, you may choose to do something different.
Jumping

The Player class supports jumping. While the player is in contact with a surface (and optionally has enough energy as defined by the PlayerData), $mvTriggerCount2 will cause the player to jump.
Jetting

The Player class includes a simple jetpack behaviour allowing characters to 'jet' upwards while jumping. The jetting behaviour can be linked to the player's energy level using datablock properties as shown below:

Example:

```plaintext
datablock PlayerData( JetPlayer )
{
  ...
  jetJumpForce = 16.0 * 90;
  jetJumpEnergyDrain = 10;
  jetMinJumpEnergy = 25;
  jetMinJumpSpeed = 20;
  jetMaxJumpSpeed = 100;
  jetJumpSurfaceAngle = 78;
}
```

This player will not be able to jet if he has less than 25 units of energy, and 10 units will be subtracted each tick.

If PlayerData::jetJumpFore is greater than zero then $mvTriggerCount1 will activate jetting.
Falling and Landing

When the player is falling they transition into the "fall" sequence. This transition doesn't occur until the player has reached a particular speed -- you don't want the fall sequence to kick in if they've just gone over a small bump. This speed threshold is set by the `PlayerData fallingSpeedThreshold` field. By default it is set to -10.0.

When the player lands there are two possible outcomes depending on how the player is set up. With the traditional method the "land" sequence has the player start from a standing position and animates into a crouch. The playback speed of this sequence is scaled based on how hard the player hits the ground. Once the land sequence finishes playing the player does a smooth transition back into the root pose (making them effectively stand up).

Starting with 1.2 there is a new method of handling landing. Here the "land" sequence starts with the player crouching on the ground and animates getting back up. This has a look of the player hitting the ground from a fall and slowly standing back up. This new method is used when the `PlayerData landSequenceTime` field is given a value greater than zero. This is the amount of time taken for the player to recover from the landing, and is also how long the land sequence will play for. As this has game play ramifications (the player may have movement constraints when landing) this timing is controlled by the datablock field rather than just the length of time of the land sequence.

Also when using the new land sequence the `PlayerData transitionToLand` flag indicates if the player should smoothly transition between the fall sequence and the land sequence. If set to false (the default) then there is no transition and the player appears to immediately go from falling to landing, which is usually the case when mirroring real life.
Air Control

The player may optionally move itself through the air while jumping or falling. This allows the player to adjust their trajectory while in the air, and is known as air control. The `PlayerData::airControl` property determines what fraction of the player's normal speed they may move while in the air. By default, air control is disabled (set to 0).
Hard Impacts

When the player hits something hard it is possible to trigger an impact (such as handled by Armor::onImpact()). The PlayerData minImpactSpeed is the threshold at which falling damage will be considered an impact. Any speed over this parameter will trigger an onImpact() call on the datablock. This allows for small falls to not cause any damage.

The PlayerData minLateralImpactSpeed is the threshold at which non-falling damage impacts will trigger the callback. This is separate from falling as you may not want a sprinting player that hits a wall to get hurt, but being thrown into a wall by an explosion will.
Dismounting

It is possible to have the player mount another object, such as a vehicle, just like any other SceneObject. While mounted, $mvTriggerCount2 will cause the player to dismount.
Triggerring a Mounted Object

A Player may have other objects mounted to it, with each mounted object assigned to a slot. These Player mounted objects are known as images. See ShapeBase::mountImage(). If there is an image mounted to slot 0, $mvTriggerCount0 will trigger it. If the player dies this trigger is automatically released.

If there is an image mounted to slot 1, $mvTriggerCount1 will trigger it. Otherwise $mvTriggerCount1 will be passed along to the image in slot 0 as an alternate fire state.
Character model

The following sequences are used by the Player object to animate the character. Not all of them are required, but a model should have at least the root, run, back and side animations. And please see the section on Sprinting above for how they are handled when not present.

root
Looping sequence played when player is standing but not moving.

run
Looping sequence played when player is running forward.

back
Looping sequence played when player is running backward.

side
Looping sequence played when player is running sideways (strafing). The sequence should depict the player moving left. If side_right is not present, this sequence will be played backwards in its place.

side_right
Looping sequence played when player is running sideways right.

sprint_root
Looping sequence played when the player is stationary but in a sprinting mode. If not present then the root sequence is used.

sprint_forward
Looping sequence played when the player is sprinting and moving forward. If not present then the run sequence is used.

sprint_backward
Looping sequence played when the player is sprinting and moving backward. If not present then the back sequence is used.

sprint_side
Looping sequence played when the player is sprinting and
moving sideways. The sequence should depict the player moving left. If crouch_right is not present, this sequence will be played backwards in its place. If not present then the side sequence is used.

**sprint_right**

Looping sequence played when the player is sprinting and moving sideways. If not present then the side_right sequence is used.

**crouch_root**

Looping sequence played when player is crouched and not moving.

**crouch_forward**

Looping sequence played when player is crouched and moving forward.

**crouch_backward**

Looping sequence played when player is crouched and moving backward.

**crouch_side**

Looping sequence played when player is crouched and moving sideways. The sequence should depict the player moving left. If crouch_right is not present, this sequence will be played backwards in its place.

**crouch_right**

Looping sequence played when player is crouched and moving sideways.

**prone_root**

Looping sequence played when player is prone (lying down) and not moving.

**prone_forward**

Looping sequence played when player is prone (lying down) and moving forward.

**prone_backward**

Looping sequence played when player is prone (lying down) and moving backward.
**swim_root**
Looping sequence played when player is swimming and not moving.

**swim_forward**
Looping sequence played when player is swimming and moving forward.

**swim_backward**
Looping sequence played when player is swimming and moving backward.

**swim_left**
Looping sequence played when player is swimming and moving left. The sequence should depict the player moving left. If swim_right is not present, this sequence will be played backwards in its place.

**swim_right**
Looping sequence played when player is swimming and moving right.

**fall**
Sequence played when player is falling.

**jump**
Sequence played when player has jumped while moving.

**standjump**
Sequence played when player has jumped from a standing start.

**land**
Sequence played when player lands after falling.

**jet**
Looping sequence played when player is jetting.

**head**
Sequence to control vertical head movement (for looking) (start=full up, end=full down).

**headside**
Sequence to control horizontal head movement (for looking) (start=full left, end=full right).
look
  Sequence to control vertical arm movement (for looking) (start=full up, end=full down).

light_recoil
  Sequence played when the player is firing a light weapon. (Based on ShapeBaseImageData)

medium_recoil
  Sequence played when player is firing a medium weapon. (Based on ShapeBaseImageData)

heavy_recoil
  Sequence played when player is firing a heavy weapon (Based on ShapeBaseImageData).

deathN
  Sequence played when player has been killed (a random one of these will play). N is an integer from 1 to 11.
Mounted Image Controlled 3rd Person Animation

A player's 3rd person action animation sequence selection may be modified based on what images are mounted on the player. When mounting a `ShapeBaseImageData`, the image's `imageAnimPrefix` field is used to control this. If this is left blank (the default) then nothing happens to the 3rd person player -- all of the sequences play as defined. If it is filled with some text (best to keep it to letters and numbers, with no spaces) then that text is added to the action animation sequence name and looked up on the player shape. For example:

A rifle `ShapeBaseImageData` is mounted to the player in slot 0. The rifle's datablock doesn't have an `imageAnimPrefix` defined, so the 3rd person player will use the standard action animation sequence names. i.e. "root", "run", "back", "crouch_root", etc.

Now a pistol `ShapeBaseImageData` is mounted to the player in slot 0. The pistol's datablock has `imageAnimPrefix` = "pistol". Now the "pistol_" (underscore is added by the system) prefix is added to each of the action animation sequence names when looking up what to play on the player's shape. So the `Player` class will look for "pistol_root", "pistol_run", "pistol_back", "pistol_crouch_root", etc. If any of these new prefixed names are not found on the player's shape, then we fall back to the standard action animation sequence names, such as "root", "run", etc.

In all of our T3D examples the player only mounts a single image. But Torque allows up to four images to be mounted at a time. When more than one image is mounted then the engine adds all of the prefixes together when searching for the action animation sequence name. If that combined name is not found then the engine starts removing prefixes starting with the highest slot down to the lowest slot. For example, if a player is holding a sword (slot 0) and a shield (slot 1) in each hand that are mounted as separate images (and with `imageAnimPrefix`'s of "sword" and "shield" respectively), then the engine will search for the following names while the player is just standing there:
• shield_sword_root
• sword_root
• root

The first one that is found in the above order will be used.

Another example: If the player has a jet pack (slot 3 with a prefix of "jetpack") and two pistols being used akimbo style (slots 1 and 0, both with a prefix of "laserpistol") with slot 2 left open for a helmet (which is skipped as it doesn't have a prefix), then the following search order would be used:

• jetpack_laserpistol_laserpistol_root
• laserpistol_laserpistol_root
• laserpistol_root
• root

Again, the first one that is found is used.

A player's 3rd person animation may also be modified by the weapon being used. In T3D 1.1 there are the three recoil sequences that may be triggered on the 3rd person player by the weapon's state. Starting with T3D 1.2 this becomes more generic (while still supporting the existing recoil sequence). When a ShapeBaseImageData state defines a stateShapeSequence, that sequence may be played on the player's shape (the new PlayerData allowImageStateAnimation field must be set to "true" as well). The new ShapeBaseImageData state stateScaleShapeSequence flag may also be used to indicate if this player animation sequence should have its playback rate scaled to the length of the image's state.

What exactly happens on the player depends on what else has been defined. First, there is the sequence name as passed in from the image. Then there is also the imageAnimPrefix as defined by the image. Finally, there is the generic script defined prefix that may be added with ShapeBase::setImageScriptAnimPrefix() -- we're using this to pass along the current pose, but it could be used for anything. Time for an example. We want to throw a grenade that we're holding (mounted in slot 0). The weapon's state that does this has
stateShapeSequence set to "throw". The grenade image itself has an imageAnimPrefix defined as "fraggrenade". Finally, the player is crouching, so Armor::onPoseChange() sets the script prefix to "crouch". The final search order goes like this:

- fraggrenade_crouch_throw
- fraggrenade_throw
- crouch_throw
- throw

The first of those sequences that is found is played as a new thread on the 3rd person player. As with recoil, only one of these 3rd person animation threads may be active at a time. If an image in another slot also asks to play a 3rd person sequence, the most recent request is what will play.
1st Person Arms

Games that have the player hold a weapon in a 1st person view often let you see the player's arms and hands holding that weapon. Rather than requiring you to build the art for all possible combinations of character arms and weapons, T3D allows you to mix and match shapes and animation sequences.

1st person arms are an optional client-side only effect and are not used on the server. The arms are a separate shape from the normal 3rd person player shape. You reference the arms using the PlayerData "shapeNameFP" array. It is an array as we support up to four mounted images therefore we support up to four arm shapes. However, for T3D 1.2 our examples only make use of a single set of arms for the first mounting slot as our example soldier holds a single weapon at a time.

As the arms are just regular DAE/DTS files they may get their animation sequences from anywhere. For the included 1.2 art path (see the soldier in the template projects) we decided that their sequences should come from the weapons themselves. This means that the weapons include all of the bones/nodes needed to animate the arms, but none of the arm geometry. If you take a look at art/shapes/actors/Soldier/FP/FP_SoldierArms.cs you'll see the external animation sequence references for each of the possible weapons.

As each weapon may require its own set of animation sequences (i.e. a different idle sequence for a pistol vs. a rifle) starting with T3D 1.2 a new ShapeBaseImageData field now exists: imagePrefixFP. If this field is defined for the mounted image then it is added to the sequence name as given in the current weapon state in the form of "prefix_sequence" (the underscore is added by the system). For example, the Lurker rifle has an imagePrefixFP of "Rifle". The Lurker's Ready state calls the idle sequence, so the arms will attempt to play the "Rifle_idle" sequence and if not found, they will play the "idle" sequence.

The advantage of having the prefix defined within the datablock and
not making it part of the sequence names referenced directly in the weapon state machine is that you can do something like this:

Example:

```plaintext
datablock ShapeBaseImageData(Pistol1Image)
{
    imageAnimPrefixFP = "Pistol1";
    ...other data here...
    ...weapon state machine here...
};

datablock ShapeBaseImageData(Pistol2Image)
{
    imageAnimPrefixFP = "Pistol2";
};
```

You could define a new pistol (Pistol2Image) that uses the exact same state machine as Pistol1Image, but could use a slightly different set of animation sequences with a prefix of "Pistol2".

As was previously discussed with 3rd person animation above, a script-based modifier may also be added when looking up the sequence name for the arms. This is currently used to pass along the player's pose so the arm's idle sequence could have a swimming motion when in the swim pose, for example. And as with images, the arms sequence name look up uses the following order to find a sequence to play, with the first one found being used:

- `ShapeBaseImageDataPrefix_ScriptPrefix_WeaponStateSequence`
- `ShapeBaseImageDataPrefix_WeaponStateSequence`
- `ScriptPrefix_WeaponStateSequence`
- `WeaponStateSequence`

Finally, the arms support an "ambient" sequence that may be used for anything and will always play, if it is defined in the arm's shape.
Example PlayerData Datablock

An example of a player datablock appears below:

Example:

datablock PlayerData(DefaultPlayerData) {
    renderFirstPerson = false;

    computeCRC = false;

    // Third person shape
    shapeFile = "art/shapes/actors/Soldier/soldier_rigged.dae";
    cameraMaxDist = 3;
    allowImageStateAnimation = true;

    // First person arms
    imageAnimPrefixFP = "soldier";
    shapeNameFP[0] = "art/shapes/actors/Soldier/FP/FP_SoldierArms.DAE";

    canObserve = 1;
    cmdCategory = "Clients";

    cameraDefaultFov = 55.0;
    cameraMinFov = 5.0;
    cameraMaxFov = 65.0;

    debrisShapeName = "art/shapes/actors/common/debris_player.dts"
    debris = playerDebris;

    throwForce = 30;

    aiAvoidThis = 1;
}
minLookAngle = "-1.2";
maxLookAngle = "1.2";
maxFreelookAngle = 3.0;

mass = 120;
drag = 1.3;
maxdrag = 0.4;
density = 1.1;
maxDamage = 100;
maxEnergy = 60;
repairRate = 0.33;
energyPerDamagePoint = 75;

rechargeRate = 0.256;

runForce = 4320;
runEnergyDrain = 0;
minRunEnergy = 0;
maxForwardSpeed = 8;
maxBackwardSpeed = 6;
maxSideSpeed = 6;

sprintForce = 4320;
sprintEnergyDrain = 0;
minSprintEnergy = 0;
maxSprintForwardSpeed = 14;
maxSprintBackwardSpeed = 8;
maxSprintSideSpeed = 6;
sprintStrafeScale = 0.25;
sprintYawScale = 0.05;
sprintPitchScale = 0.05;
sprintCanJump = true;
crouchForce = 405;
maxCrouchForwardSpeed = 4.0;
maxCrouchBackwardSpeed = 2.0;
maxCrouchSideSpeed = 2.0;

maxUnderwaterForwardSpeed = 8.4;
maxUnderwaterBackwardSpeed = 7.8;
maxUnderwaterSideSpeed = 7.8;

jumpForce = "747";
jumpEnergyDrain = 0;
minJumpEnergy = 0;
jumpDelay = "15";
airControl = 0.3;

fallingSpeedThreshold = -6.0;

landSequenceTime = 0.33;
transitionToLand = false;
recoverDelay = 0;
recoverRunForceScale = 0;

minImpactSpeed = 10;
minLateralImpactSpeed = 20;
speedDamageScale = 0.4;

boundingBox = "0.65 0.75 1.85";
crouchBoundingBox = "0.65 0.75 1.3";
swimBoundingBox = "1 2 2";
pickupRadius = 1;

// Damage location details
boxHeadPercentage = 0.83;
boxTorsoPercentage = 0.49;
boxHeadLeftPercentage = 0.30;
boxHeadRightPercentage = 0.60;
boxHeadBackPercentage = 0.30;
boxHeadFrontPercentage = 0.60;

// Foot Prints
decalOffset = 0.25;

footPuffEmitter = "LightPuffEmitter";
footPuffNumParts = 10;
footPuffRadius = "0.25";

dustEmitter = "LightPuffEmitter";

splash = PlayerSplash;
splashVelocity = 4.0;
splashAngle = 67.0;
splashFreqMod = 300.0;
splashVelEpsilon = 0.60;
bubbleEmitTime = 0.4;
splashEmitter[0] = PlayerWakeEmitter;
splashEmitter[1] = PlayerFoamEmitter;
splashEmitter[2] = PlayerBubbleEmitter;
mediumSplashSoundVelocity = 10.0;
hardSplashSoundVelocity = 20.0;
exitSplashSoundVelocity = 5.0;

// Controls over slope of runnable/jumpable surfaces
runSurfaceAngle = 38;
jumpSurfaceAngle = 80;
maxStepHeight = 0.35; // two meters
minJumpSpeed = 20;
maxJumpSpeed = 30;

horizMaxSpeed = 68;
horizResistSpeed = 33;
horizResistFactor = 0.35;

upMaxSpeed = 80;
upResistSpeed = 25;
upResistFactor = 0.3;

footstepSplashHeight = 0.35;

// Footstep Sounds
FootSoftSound = FootLightSoftSound;
FootHardSound = FootLightHardSound;
FootMetalSound = FootLightMetalSound;
FootSnowSound = FootLightSnowSound;
FootShallowSound = FootLightShallowSound;
FootWadingSound = FootLightWadingSound;
FootUnderwaterSound = FootLightUnderwaterSound;

FootBubblesSound = FootLightBubblesSound;
movingBubblesSound = ArmorMoveBubblesSound;
waterBreathSound = WaterBreathMaleSound;

impactSoftSound = ImpactLightSoftSound;
impactHardSound = ImpactLightHardSound;
impactMetalSound = ImpactLightMetalSound;
impactSnowSound = ImpactLightSnowSound;

impactWaterEasy = ImpactLightWaterEasySound;
impactWaterMedium = ImpactLightWaterMediumSound;
impactWaterHard = ImpactLightWaterHard;

groundImpactMinSpeed = "45";
groundImpactShakeFreq = "4.0 4.0 4.0"
groundImpactShakeAmp = "1.0 1.0 1.0"
groundImpactShakeDuration = 0.8;
groundImpactShakeFalloff = 10.0;

exitingWater = ExitingWaterLightSound;

observeParameters = "0.5 4.5 4.5";
class = "armor";

cameraMinDist = "0";
DecalData = "PlayerFootprint";

// Allowable Inventory Items
mainWeapon = Lurker;

maxInv[Lurker] = 1;
maxInv[LurkerClip] = 20;

maxInv[LurkerGrenadeLauncher] = 1;
maxInv[LurkerGrenadeAmmo] = 20;

maxInv[Ryder] = 1;
maxInv[RyderClip] = 10;

maxInv[ProxMine] = 5;
maxInv[DeployableTurret] = 5;

// available skins (see materials.cs in
availableSkins = "base DarkBlue DarkGreen LightGreen Orange Red Teal Violet Yellow";
void Player::allowAllPoses()

Allow all poses a chance to occur.

This method resets any poses that have manually been blocked from occurring. This includes the regular pose states such as sprinting, crouch, being prone and swimming. It also includes being able to jump and jet jump. While this is allowing these poses to occur it doesn't mean that they all can due to other conditions. We're just not manually blocking them from being allowed.

See also:
- allowJumping()
- allowJetJumping()
- allowSprinting()
- allowCrouching()
- allowProne()
- allowProne()

void Player::allowCrouching(bool state)

Set if the Player is allowed to crouch.

The default is to allow crouching unless there are other environmental concerns that prevent it. This method is mainly used to explicitly disallow crouching at any time.

Parameters:
- state Set to true to allow crouching, false to disable it.

See also:
- allowAllPoses()
void Player::allowJetJumping(bool state)

Set if the Player is allowed to jet jump.

The default is to allow jet jumping unless there are other environmental concerns that prevent it. This method is mainly used to explicitly disallow jet jumping at any time.

Parameters:

state Set to true to allow jet jumping, false to disable it.

See also:

allowAllPoses()

void Player::allowJumping(bool state)

Set if the Player is allowed to jump.

The default is to allow jumping unless there are other environmental concerns that prevent it. This method is mainly used to explicitly disallow jumping at any time.

Parameters:

state Set to true to allow jumping, false to disable it.

See also:

allowAllPoses()

void Player::allowProne(bool state)

Set if the Player is allowed to go prone.

The default is to allow being prone unless there are other environmental concerns that prevent it. This method is mainly used to explicitly disallow going prone at any time.
Parameters:

state  Set to true to allow being prone, false to disable it.

See also:

allowAllPoses()

void Player::allowSprinting(bool state)

Set if the Player is allowed to sprint.

The default is to allow sprinting unless there are other environmental concerns that prevent it. This method is mainly used to explicitly disallow sprinting at any time.

Parameters:

state  Set to true to allow sprinting, false to disable it.

See also:

allowAllPoses()

void Player::allowSwimming(bool state)

Set if the Player is allowed to swim.

The default is to allow swimming unless there are other environmental concerns that prevent it. This method is mainly used to explicitly disallow swimming at any time.

Parameters:

state  Set to true to allow swimming, false to disable it.

See also:

allowAllPoses()
bool Player::checkDismountPoint(Point3F oldPos, Point3F pos)

Check if it is safe to dismount at this position.

Internally this method casts a ray from oldPos to pos to determine if it hits the terrain, an interior object, a water object, another player, a static shape, a vehicle (excluding the one currently mounted), or physical zone. If this ray is in the clear, then the player's bounding box is also checked for a collision at the pos position. If this displaced bounding box is also in the clear, then checkDismountPoint() returns true.

**Parameters:**

- **oldPos** The player's current position
- **pos** The dismount position to check

**Returns:**

True if the dismount position is clear, false if not

**Note:**

The player must be already mounted for this method to not assert.

void Player::clearControlObject()

Clears the player's current control object.

Returns control to the player. This internally calls Player::setControlObject(0).

**Example:**

```
%player.clearControlObject();
echo(%player.getControlObject()); // <-- Re
```
%player.setControlObject(%vehicle);
echo(%player.getControlObject()); //<-- Returns %vehicle, player controls the vehicle now.

Note:
If the player does not have a control object, the player will receive all moves from its GameConnection. If you're looking to remove control from the player itself (i.e. stop sending moves to the player) use GameConnection::setControlObject() to transfer control to another object, such as a camera.

See also:
setControlObject()
getControlObject()
GameConnection::setControlObject()

int Player::getControlObject()
Get the current object we are controlling.

Returns:
ID of the ShapeBase object we control, or 0 if not controlling an object.

See also:
setControlObject()
clearControlObject()

string Player::getDamageLocation(Point3F pos)
Get the named damage location and modifier for a given world position.

the Player object can simulate different hit locations based on a pre-defined set of PlayerData defined percentages. These hit
percentages divide up the Player's bounding box into different regions. The diagram below demonstrates how the various PlayerData properties split up the bounding volume:

While you may pass in any world position and getDamageLocation() will provide a best-fit location, you should be aware that this can produce some interesting results. For example, any position that is above PlayerData::boxHeadPercentage will be considered a 'head' hit, even if the world position is high in the sky.
Therefore it may be wise to keep the passed in point to somewhere on the surface of, or within, the Player’s bounding volume.

**Note:**
This method will not return an accurate location when the player is prone or swimming.

**Parameters:**

`pos` A world position for which to retrieve a body region on this player.

**Returns:**
a string containing two words (space separated strings), where the first is a location and the second is a modifier.

Possible locations:

- head
- torso
- legs

Head modifiers:

- left_back
- middle_back
- right_back
- left_middle
- middle_middle
- right_middle
- left_front
- middle_front
- right_front

Legs/Torso modifiers:

- front_left
- front_right
- back_left
• back_right

See also:

PlayerData::boxHeadPercentage
PlayerData::boxHeadFrontPercentage
PlayerData::boxHeadBackPercentage
PlayerData::boxHeadLeftPercentage
PlayerData::boxHeadRightPercentage
PlayerData::boxTorsoPercentage

int Player::getNumDeathAnimations()

Get the number of death animations available to this player.

Death animations are assumed to be named death1-N using consecutive indices.

string Player::getPose()

Get the name of the player's current pose.

The pose is one of the following:

• Stand - Standard movement pose.
• Sprint - Sprinting pose.
• Crouch - Crouch pose.
• Prone - Prone pose.
• Swim - Swimming pose.

Returns:

The current pose; one of: "Stand", "Sprint", "Crouch", "Prone", "Swim"

string Player::getState()
Get the name of the player's current state.

The state is one of the following:

- **Dead** - The Player is dead.
- **Mounted** - The Player is mounted to an object such as a vehicle.
- **Move** - The Player is free to move. The usual state.
- **Recover** - The Player is recovering from a fall. See `PlayerData::recoverDelay`.

**Returns:**

The current state; one of: "Dead", "Mounted", "Move", "Recover"

```cpp
bool Player::setActionThread(string name,  
    bool  hold = false,  
    bool  fsp = true
)
```

Set the main action sequence to play for this player.

**Parameters:**

- **name** Name of the action sequence to set
  Set to false to get a callback on the datablock when
- **hold** the sequence ends (`PlayerData::animationDone()`).
  When set to true no callback is made.
- **fsp** True if first person and none of the spine nodes in the
  shape should animate. False will allow the shape's
  spine nodes to animate.

**Returns:**

True if succesful, false if failed

**Note:**

The spine nodes for the Player's shape are named as follows:
- Bip01 Pelvis
- Bip01 Spine
- Bip01 Spine1
- Bip01 Spine2
- Bip01 Neck
- Bip01 Head

You cannot use `setActionThread()` to have the Player play one of the motion determined action animation sequences. These sequences are chosen based on how the Player moves and the Player's current pose. The names of these sequences are:

- root
- run
- side
- side_right
- crouch_root
- crouch_forward
- crouch_backward
- crouch_side
- crouch_right
- prone_root
- prone_forward
- prone_backward
- swim_root
- swim_forward
- swim_backward
- swim_left
- swim_right
- fall
- jump
- standjump
- land
- jet

If the player moves in any direction then the animation sequence set using this method will be cancelled and the chosen motion-based sequence will take over. This makes great for times when the Player cannot move, such as when mounted, or when it
doesn't matter if the action sequence changes, such as waving and saluting.

Example:

```cpp
// Place the player in a sitting position
%player.setActionThread( "sitting", true,

bool Player::setArmThread(string name )

Set the sequence that controls the player's arms (dynamically adjusted to match look direction).

Parameters:

  name  Name of the sequence to play on the player's arms.

Returns:

  true if successful, false if failed.

Note:

  By default the 'look' sequence is used, if available.

bool Player::setControlObject(ShapeBase obj )

Set the object to be controlled by this player.

It is possible to have the moves sent to the Player object from the GameConnection to be passed along to another object. This happens, for example when a player is mounted to a vehicle. The move commands pass through the Player and on to the vehicle (while the player remains stationary within the vehicle). With setControlObject() you can have the Player pass along its moves to any object. One possible use is for a player to move a remote controlled vehicle. In this case the player does not mount the vehicle directly, but still wants to be able to control it.
Parameters:

obj Object to control with this player

Returns:

True if the object is valid, false if not

See also:

getControlObject()
clearControlObject()
GameConnection::setControlObject()
Member Data Documentation

int Player::crouchTrigger [static]

The move trigger index used for player crouching.

int Player::imageTrigger0 [static]

The move trigger index used to trigger mounted image 0.

int Player::imageTrigger1 [static]

The move trigger index used to trigger mounted image 1 or alternate fire on mounted image 0.

int Player::jumpJetTrigger [static]

The move trigger index used for player jump jetting.

int Player::jumpTrigger [static]

The move trigger index used for player jumping.

float Player::maxImpulseVelocity [static]

The maximum velocity allowed due to a single impulse.

int Player::maxPredictionTicks [static]
Maximum number of ticks to predict on the client from the last known move obtained from the server.

```cpp
int Player::maxWarpTicks [static]
```

When a warp needs to occur due to the client being too far off from the server, this is the maximum number of ticks we'll allow the client to warp to catch up.

```cpp
float Player::minWarpTicks [static]
```

Fraction of tick at which instant warp occurs on the client.

```cpp
int Player::proneTrigger [static]
```

The move trigger index used for player prone pose.

```cpp
bool Player::renderCollision [static]
```

Determines if the player's collision mesh should be rendered. This is mainly used for the tools and debugging.

```cpp
bool Player::renderMyItems [static]
```

Determines if mounted shapes are rendered or not. Used on the client side to disable the rendering of all `Player` mounted objects. This is mainly used for the tools or debugging.
**bool** Player::renderMyPlayer [static]

Determines if the player is rendered or not.

Used on the client side to disable the rendering of all **Player** objects. This is mainly for the tools or debugging.

**int** Player::sprintTrigger [static]

The move trigger index used for player sprinting.

**int** Player::vehicleDismountTrigger [static]

The move trigger index used to dismount player.

Copyright © GarageGames, LLC. All Rights Reserved.
PlayerData Class Reference
[Game Objects]

Defines properties for a Player object. More...

Inheritance diagram for PlayerData:

List of all members.
### Public Member Functions

**Callbacks**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void <code>animationDone</code> (Player obj)</td>
<td>Called on the server when a scripted animation completes.</td>
</tr>
<tr>
<td>void <code>doDismount</code> (Player obj)</td>
<td>Called when attempting to dismount the player from a vehicle.</td>
</tr>
<tr>
<td>void <code>onEnterLiquid</code> (Player obj, float coverage, string type)</td>
<td>Called when the player enters a liquid.</td>
</tr>
<tr>
<td>void <code>onEnterMissionArea</code> (Player obj)</td>
<td>Called when the player enters the mission area.</td>
</tr>
<tr>
<td>void <code>onLeaveLiquid</code> (Player obj, string type)</td>
<td>Called when the player leaves a liquid.</td>
</tr>
<tr>
<td>void <code>onLeaveMissionArea</code> (Player obj)</td>
<td>Called when the player leaves the mission area.</td>
</tr>
<tr>
<td>void <code>onPoseChange</code> (Player obj, string oldPose, string newPose)</td>
<td>Called when the player changes poses.</td>
</tr>
<tr>
<td>void <code>onStartSprintMotion</code> (Player obj)</td>
<td>Called when the player starts moving while in a Sprint pose.</td>
</tr>
<tr>
<td>void <code>onStartSwim</code> (Player obj)</td>
<td>Called when the player starts swimming.</td>
</tr>
<tr>
<td>void <code>onStopSprintMotion</code> (Player obj)</td>
<td>Called when the player stops moving while in a Sprint pose.</td>
</tr>
<tr>
<td>void <code>onStopSwim</code> (Player obj)</td>
<td>Called when the player stops swimming.</td>
</tr>
</tbody>
</table>
## Public Attributes

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>maxTimeScale</code></td>
<td>Maximum time scale for action animations.</td>
</tr>
<tr>
<td>float</td>
<td><code>pickupRadius</code></td>
<td>Radius around the player to collide with Items in the scene (on server).</td>
</tr>
</tbody>
</table>

## Movement: Jumping

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>airControl</code></td>
<td>Amount of movement control the player has when in the air.</td>
</tr>
<tr>
<td>int</td>
<td><code>jumpDelay</code></td>
<td>Delay time in number of ticks ticks between jumps.</td>
</tr>
<tr>
<td>float</td>
<td><code>jumpEnergyDrain</code></td>
<td>Energy level drained each time the player jumps.</td>
</tr>
<tr>
<td>float</td>
<td><code>jumpForce</code></td>
<td>Force used to accelerate the player when a jump is initiated.</td>
</tr>
<tr>
<td>float</td>
<td><code>jumpSurfaceAngle</code></td>
<td>Angle from vertical (in degrees) where the player can jump.</td>
</tr>
<tr>
<td>bool</td>
<td><code>jumpTowardsNormal</code></td>
<td>Controls the direction of the jump impulse.</td>
</tr>
<tr>
<td>float</td>
<td><code>maxJumpSpeed</code></td>
<td>Maximum vertical speed before the player can no longer jump.</td>
</tr>
<tr>
<td>float</td>
<td><code>minJumpEnergy</code></td>
<td>Minimum energy level required to jump.</td>
</tr>
<tr>
<td>Type</td>
<td>Variable</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>float</td>
<td>minJumpSpeed</td>
<td>Minimum speed needed to jump.</td>
</tr>
<tr>
<td>Third Person</td>
<td>allowImageStateAnimation</td>
<td>Allow mounted images to request a sequence be played on the Player.</td>
</tr>
<tr>
<td>caseString</td>
<td>imageAnimPrefix</td>
<td>Optional prefix to all mounted image animation sequences in third person.</td>
</tr>
<tr>
<td>Collision</td>
<td>boundingBox</td>
<td>Size of the bounding box used by the player for collision.</td>
</tr>
<tr>
<td>float</td>
<td>boxHeadBackPercentage</td>
<td>Percentage of the player's bounding box depth that represents the back side of the head.</td>
</tr>
<tr>
<td>float</td>
<td>boxHeadFrontPercentage</td>
<td>Percentage of the player's bounding box depth that represents the front side of the head.</td>
</tr>
<tr>
<td>float</td>
<td>boxHeadLeftPercentage</td>
<td>Percentage of the player's bounding box width that represents the left side of the head.</td>
</tr>
<tr>
<td>float</td>
<td>boxHeadPercentage</td>
<td>Percentage of the player's bounding box height that represents the head.</td>
</tr>
<tr>
<td>float</td>
<td>boxHeadRightPercentage</td>
<td>Percentage of the player's bounding box width that represents the right side of the head.</td>
</tr>
</tbody>
</table>
head.

<table>
<thead>
<tr>
<th>float</th>
<th>boxTorsoPercentage</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Percentage of the player's bounding box height that represents the torso.</td>
</tr>
</tbody>
</table>

**Point3F crouchBoundingBox**
Collision bounding box used when the player is crouching.

**Point3F proneBoundingBox**
Collision bounding box used when the player is prone (laying down).

**Point3F swimBoundingBox**
Collision bounding box used when the player is swimming.

**Interaction: Splashes**

<table>
<thead>
<tr>
<th>float</th>
<th>bubbleEmitTime</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time in seconds to generate bubble particles after entering the water.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>exitSplashSoundVelocity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Minimum velocity when leaving the water for the exitingWater sound to play.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>footstepSplashHeight</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Water coverage level to choose between FootShallowSound and FootWadingSound.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>hardSplashSoundVelocity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Minimum velocity when entering the water for choosing between the impactWaterMedium and impactWaterHard sound to play.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>mediumSplashSoundVelocity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Minimum velocity when entering the water for choosing between the impactWaterEasy and impactWaterMedium sounds to play.</td>
</tr>
</tbody>
</table>
### SplashData

**Splash**

**SplashData** datablock used to create splashes when the player moves through water.

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>splashAngle</td>
<td>Maximum angle (in degrees) from pure vertical movement in water to generate splashes.</td>
</tr>
</tbody>
</table>

### ParticleEmitterData

**splashEmitter** [3]

Particle emitters used to generate splash particles.

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>splashFreqMod</td>
<td>Multiplied by speed to determine the number of splash particles to generate.</td>
</tr>
<tr>
<td>float</td>
<td>splashVelEpsilon</td>
<td>Minimum speed to generate splash particles.</td>
</tr>
<tr>
<td>float</td>
<td>splashVelocity</td>
<td>Minimum velocity when moving through water to generate splashes.</td>
</tr>
</tbody>
</table>

### Movement: Crouching

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>crouchForce</td>
<td>Force used to accelerate the player when crouching.</td>
</tr>
<tr>
<td>float</td>
<td>maxCrouchBackwardSpeed</td>
<td>Maximum backward speed when crouching.</td>
</tr>
<tr>
<td>float</td>
<td>maxCrouchForwardSpeed</td>
<td>Maximum forward speed when crouching.</td>
</tr>
<tr>
<td>float</td>
<td>maxCrouchSideSpeed</td>
<td>Maximum sideways speed when crouching.</td>
</tr>
</tbody>
</table>

### Interaction: Footsteps
### Decal Data
- **DecalData**
  Decal to place on the ground for player footsteps.

### Float Data
- **decalOffset**
  Distance from the center of the model to the right foot.

### Particle Emitter Data
- **dustEmitter**
  Emitter used to generate dust particles.
- **footPuffEmitter**
  Particle emitter used to generate footpuffs (particles created as the player walks along the ground).

### Integer Data
- **footPuffNumParts**
  Number of footpuff particles to generate each step.

### Float Data
- **footPuffRadius**
  Particle creation radius for footpuff particles.

### Interaction: Sounds

<table>
<thead>
<tr>
<th>SFXTrack</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>exitingWater</strong></td>
<td>Sound to play when exiting the water with velocity ( \geq ) exitSplashSoundVelocity.</td>
</tr>
<tr>
<td><strong>FootBubblesSound</strong></td>
<td>Sound to play when walking in water and coverage equals 1.0 (fully underwater).</td>
</tr>
<tr>
<td><strong>FootHardSound</strong></td>
<td>Sound to play when walking on a surface with Material ( \text{footstepSoundId} ) 1.</td>
</tr>
<tr>
<td><strong>FootMetalSound</strong></td>
<td>Sound to play when walking on a surface with Material ( \text{footstepSoundId} ) 2.</td>
</tr>
<tr>
<td><strong>FootShallowSound</strong></td>
<td>Sound to play when walking on a surface with Material ( \text{footstepSoundId} ) 1.</td>
</tr>
<tr>
<td>SFXTrack</td>
<td>Sound Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>FootSnowSound</td>
<td>Sound to play when walking on a surface with Material footstepSoundId 3.</td>
</tr>
<tr>
<td>FootSoftSound</td>
<td>Sound to play when walking on a surface with Material footstepSoundId 0.</td>
</tr>
<tr>
<td>FootUnderwaterSound</td>
<td>Sound to play when walking in water and coverage equals 1.0 (fully underwater).</td>
</tr>
<tr>
<td>FootWadingSound</td>
<td>Sound to play when walking in water and coverage is less than 1, but &gt; footSplashHeight.</td>
</tr>
<tr>
<td>impactHardSound</td>
<td>Sound to play after falling on a surface with Material footstepSoundId 1.</td>
</tr>
<tr>
<td>impactMetalSound</td>
<td>Sound to play after falling on a surface with Material footstepSoundId 2.</td>
</tr>
<tr>
<td>impactSnowSound</td>
<td>Sound to play after falling on a surface with Material footstepSoundId 3.</td>
</tr>
<tr>
<td>impactSoftSound</td>
<td>Sound to play after falling on a surface with Material footstepSoundId 0.</td>
</tr>
<tr>
<td>impactWaterEasy</td>
<td>Sound to play when entering the water with velocity &lt; mediumSplashSoundVelocity.</td>
</tr>
<tr>
<td>impactWaterHard</td>
<td>Sound to play when entering the water with</td>
</tr>
</tbody>
</table>
velocity $\geq$ hardSplashSoundVelocity.

**SFXTrack impactWaterMedium**
Sound to play when entering the water with velocity $\geq$ mediumSplashSoundVelocity and $<$ hardSplashSoundVelocity.

**SFXTrack movingBubblesSound**
Sound to play when in water and coverage equals 1.0 (fully underwater).

**SFXTrack waterBreathSound**
Sound to play when in water and coverage equals 1.0 (fully underwater).

**Falling**

**float fallingSpeedThreshold**
Downward speed at which we consider the player falling.

**float landSequenceTime**
Time of land sequence play back when using new recover system.

**int recoverDelay**
Number of ticks for the player to recover from falling.

**float recoverRunForceScale**
Scale factor applied to runForce while in the recover state.

**bool transitionToLand**
When going from a fall to a land, should we transition between the two.

**Camera**

*The settings used by the shape when it is the camera.*
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>firstPersonShadows</td>
</tr>
<tr>
<td></td>
<td>Forces shadows to be rendered in first person when renderFirstPerson is disabled. Defaults to false.</td>
</tr>
<tr>
<td>float</td>
<td>maxFreelookAngle</td>
</tr>
<tr>
<td></td>
<td>Defines the maximum left and right angles (in radians) the player can look in freelook mode.</td>
</tr>
<tr>
<td>float</td>
<td>maxLookAngle</td>
</tr>
<tr>
<td></td>
<td>Highest angle (in radians) the player can look.</td>
</tr>
<tr>
<td>float</td>
<td>minLookAngle</td>
</tr>
<tr>
<td></td>
<td>Lowest angle (in radians) the player can look.</td>
</tr>
<tr>
<td>bool</td>
<td>renderFirstPerson</td>
</tr>
<tr>
<td></td>
<td>Flag controlling whether to render the player shape in first person view.</td>
</tr>
</tbody>
</table>

**Interaction: Ground Impact**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>groundImpactMinSpeed</td>
</tr>
<tr>
<td></td>
<td>Minimum falling impact speed to apply damage and initiate the camera shaking effect.</td>
</tr>
<tr>
<td>Point3F</td>
<td>groundImpactShakeAmp</td>
</tr>
<tr>
<td></td>
<td>Amplitude of the camera shake effect after falling.</td>
</tr>
<tr>
<td>float</td>
<td>groundImpactShakeDuration</td>
</tr>
<tr>
<td></td>
<td>Duration (in seconds) of the camera shake effect after falling.</td>
</tr>
<tr>
<td>float</td>
<td>groundImpactShakeFalloff</td>
</tr>
<tr>
<td></td>
<td>Falloff factor of the camera shake effect after falling.</td>
</tr>
<tr>
<td>Point3F</td>
<td>groundImpactShakeFreq</td>
</tr>
<tr>
<td></td>
<td>Frequency of the camera shake effect after falling.</td>
</tr>
</tbody>
</table>
### Movement

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>horizMaxSpeed</td>
<td>Maximum horizontal speed.</td>
</tr>
<tr>
<td>float</td>
<td>horizResistFactor</td>
<td>Factor of resistance once horizResistSpeed has been reached.</td>
</tr>
<tr>
<td>float</td>
<td>horizResistSpeed</td>
<td>Horizontal speed at which resistance will take place.</td>
</tr>
<tr>
<td>float</td>
<td>maxBackwardSpeed</td>
<td>Maximum backward speed when running.</td>
</tr>
<tr>
<td>float</td>
<td>maxForwardSpeed</td>
<td>Maximum forward speed when running.</td>
</tr>
<tr>
<td>float</td>
<td>maxSideSpeed</td>
<td>Maximum sideways speed when running.</td>
</tr>
<tr>
<td>float</td>
<td>maxStepHeight</td>
<td>Maximum height the player can step up.</td>
</tr>
<tr>
<td>float</td>
<td>minImpactSpeed</td>
<td>Minimum impact speed to apply falling damage.</td>
</tr>
<tr>
<td>float</td>
<td>minLaterallImpactSpeed</td>
<td>Minimum impact speed to apply non-falling damage.</td>
</tr>
<tr>
<td>float</td>
<td>minRunEnergy</td>
<td>Minimum energy level required to run or swim.</td>
</tr>
<tr>
<td>float</td>
<td>runEnergyDrain</td>
<td>Energy value drained each tick that the player is moving.</td>
</tr>
<tr>
<td>float</td>
<td>runForce</td>
<td></td>
</tr>
</tbody>
</table>
Force used to accelerate the player when running.

<table>
<thead>
<tr>
<th><strong>float</strong></th>
<th><strong>runSurfaceAngle</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maximum angle from vertical (in degrees) the player can run up.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong></th>
<th><strong>upMaxSpeed</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maximum upwards speed.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong></th>
<th><strong>upResistFactor</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Factor of resistance once upResistSpeed has been reached.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong></th>
<th><strong>upResistSpeed</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Upwards speed at which resistance will take place.</td>
</tr>
</tbody>
</table>

**First Person Arms**

<table>
<thead>
<tr>
<th><strong>caseString</strong></th>
<th><strong>imageAnimPrefixFP</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Optional prefix to all mounted image animation sequences in first person.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>filename</strong></th>
<th><strong>shapeNameFP [4]</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>File name of this player's shape that will be used in conjunction with the corresponding mounted image.</td>
</tr>
</tbody>
</table>

**Movement: Jetting**

<table>
<thead>
<tr>
<th><strong>float</strong></th>
<th><strong>jetJumpEnergyDrain</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Energy level drained each time the player jet jumps.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong></th>
<th><strong>jetJumpForce</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Force used to accelerate the player when a jet jump is initiated.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong></th>
<th><strong>jetJumpSurfaceAngle</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Angle from vertical (in degrees) where the</td>
</tr>
</tbody>
</table>
player can jet jump.

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>jetMaxJumpSpeed</td>
<td>Maximum vertical speed before the player can no longer jet jump.</td>
</tr>
<tr>
<td>float</td>
<td>jetMinJumpEnergy</td>
<td>Minimum energy level required to jet jump.</td>
</tr>
<tr>
<td>float</td>
<td>jetMinJumpSpeed</td>
<td>Minimum speed needed to jet jump.</td>
</tr>
</tbody>
</table>

**Movement: Prone**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>maxProneBackwardSpeed</td>
<td>Maximum backward speed when prone (laying down).</td>
</tr>
<tr>
<td>float</td>
<td>maxProneForwardSpeed</td>
<td>Maximum forward speed when prone (laying down).</td>
</tr>
<tr>
<td>float</td>
<td>maxProneSideSpeed</td>
<td>Maximum sideways speed when prone (laying down).</td>
</tr>
<tr>
<td>float</td>
<td>proneForce</td>
<td>Force used to accelerate the player when prone (laying down).</td>
</tr>
</tbody>
</table>

**Movement: Sprinting**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>maxSprintBackwardSpeed</td>
<td>Maximum backward speed when sprinting.</td>
</tr>
<tr>
<td>float</td>
<td>maxSprintForwardSpeed</td>
<td>Maximum forward speed when sprinting.</td>
</tr>
<tr>
<td>float</td>
<td>maxSprintSideSpeed</td>
<td>Maximum sideways speed when sprinting.</td>
</tr>
<tr>
<td>Type</td>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>float</td>
<td>minSprintEnergy</td>
<td>Minimum energy level required to sprint.</td>
</tr>
<tr>
<td>bool</td>
<td>sprintCanJump</td>
<td>Can the player jump while sprinting.</td>
</tr>
<tr>
<td>float</td>
<td>sprintEnergyDrain</td>
<td>Energy value drained each tick that the player is sprinting.</td>
</tr>
<tr>
<td>float</td>
<td>sprintForce</td>
<td>Force used to accelerate the player when sprinting.</td>
</tr>
<tr>
<td>float</td>
<td>sprintPitchScale</td>
<td>Amount to scale pitch motion while sprinting.</td>
</tr>
<tr>
<td>float</td>
<td>sprintStrafeScale</td>
<td>Amount to scale strafing motion vector while sprinting.</td>
</tr>
<tr>
<td>float</td>
<td>sprintYawScale</td>
<td>Amount to scale yaw motion while sprinting.</td>
</tr>
</tbody>
</table>

**Movement: Swimming**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>maxUnderwaterBackwardSpeed</td>
<td>Maximum backward speed when underwater.</td>
</tr>
<tr>
<td>float</td>
<td>maxUnderwaterForwardSpeed</td>
<td>Maximum forward speed when underwater.</td>
</tr>
<tr>
<td>float</td>
<td>maxUnderwaterSideSpeed</td>
<td>Maximum sideways speed when underwater.</td>
</tr>
<tr>
<td>float</td>
<td>swimForce</td>
<td>Force used to accelerate the player when swimming.</td>
</tr>
</tbody>
</table>

**Physics**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>physicsPlayerType</td>
<td></td>
</tr>
<tr>
<td>Specifies the type of physics used by the player.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

Defines properties for a Player object.

See also:

Player
Member Function Documentation

```cpp
void PlayerData::animationDone(Player obj)
```

Called on the server when a scripted animation completes.

**Parameters:**

- `obj` The `Player` object

**See also:**

- `Player::setActionThread()` for setting a scripted animation and its 'hold' parameter to determine if this callback is used.

```cpp
void PlayerData::doDismount(Player obj)
```

Called when attempting to dismount the player from a vehicle.

It is up to the `doDismount()` method to actually perform the dismount. Often there are some conditions that prevent this, such as the vehicle moving too fast.

**Parameters:**

- `obj` The `Player` object

```cpp
void PlayerData::onEnterLiquid(Player obj, float coverage, string type)
```

Called when the player enters a liquid.

**Parameters:**

- `obj` The `Player` object
coverage Percentage of the player's bounding box covered by the liquid

*type* The type of liquid the player has entered

```cpp
void PlayerData::onEnterMissionArea(Player obj)
```

Called when the player enters the mission area.

**Parameters:**

*obj* The *Player* object

**See also:**

MissionArea

```cpp
void PlayerData::onLeaveLiquid(Player obj, string type)
```

Called when the player leaves a liquid.

**Parameters:**

*obj* The *Player* object

*type* The type of liquid the player has left

```cpp
void PlayerData::onLeaveMissionArea(Player obj)
```

Called when the player leaves the mission area.

**Parameters:**

*obj* The *Player* object

**See also:**

MissionArea
void PlayerData::onPoseChange(Player obj, string oldPose, string newPose)

Called when the player changes poses.

Parameters:
obj The Player object
oldPose The pose the player is switching from.
newPose The pose the player is switching to.

void PlayerData::onStartSprintMotion(Player obj)

Called when the player starts moving while in a Sprint pose.

Parameters:
obj The Player object

void PlayerData::onStartSwim(Player obj)

Called when the player starts swimming.

Parameters:
obj The Player object

void PlayerData::onStopSprintMotion(Player obj)

Called when the player stops moving while in a Sprint pose.

Parameters:
void PlayerData::onStopSwim(
      Player obj
    )

Called when the player stops swimming.

**Parameters:**

- `obj` The `Player` object
## Member Data Documentation

### float PlayerData::airControl

Amount of movement control the player has when in the air.

This is applied as a multiplier to the player's x and y motion.

### bool PlayerData::allowImageStateAnimation

Allow mounted images to request a sequence be played on the Player.

When true a new thread is added to the player to allow for mounted images to request a sequence be played on the player through the image's state machine. It is only optional so that we don't create a TSThread on the player if we don't need to.

### Point3F PlayerData::boundingBox

Size of the bounding box used by the player for collision.

Dimensions are given as "width depth height".

### float PlayerData::boxHeadBackPercentage

Percentage of the player's bounding box depth that represents the back side of the head.

Used when computing the damage location.

**See also:**

Player::getDamageLocation
float PlayerData::boxHeadFrontPercentage

Percentage of the player's bounding box depth that represents the front side of the head.

Used when computing the damage location.

See also:
    Player::getDamageLocation

float PlayerData::boxHeadLeftPercentage

Percentage of the player's bounding box width that represents the left side of the head.

Used when computing the damage location.

See also:
    Player::getDamageLocation

float PlayerData::boxHeadPercentage

Percentage of the player's bounding box height that represents the head.

Used when computing the damage location.

See also:
    Player::getDamageLocation

float PlayerData::boxHeadRightPercentage

Percentage of the player's bounding box width that represents the
right side of the head.

Used when computing the damage location.

See also:

Player::getDamageLocation

float PlayerData::boxTorsoPercentage

Percentage of the player's bounding box height that represents the torso.

Used when computing the damage location.

See also:

Player::getDamageLocation

float PlayerData::bubbleEmitTime

Time in seconds to generate bubble particles after entering the water.

Point3F PlayerData::crouchBoundingBox

Collision bounding box used when the player is crouching.

See also:

boundingBox

float PlayerData::crouchForce

Force used to accelerate the player when crouching.
DecalData PlayerData::DecalData

Decal to place on the ground for player footsteps.

float PlayerData::decalOffset

Distance from the center of the model to the right foot.

While this defines the distance to the right foot, it is also used to place the left foot decal as well. Just on the opposite side of the player.

ParticleEmitterData PlayerData::dustEmitter

Emitter used to generate dust particles.

**Note:**

Currently unused.

SFXTrack PlayerData::exitingWater

Sound to play when exiting the water with velocity >= exitSplashSoundVelocity.

**See also:**

exitSplashSoundVelocity

float PlayerData::exitSplashSoundVelocity

Minimum velocity when leaving the water for the exitingWater sound to play.
**See also:**

- exitingWater

---

**float PlayerData::fallingSpeedThreshold**

Downward speed at which we consider the player falling.

---

**bool PlayerData::firstPersonShadows**

Forces shadows to be rendered in first person when renderFirstPerson is disabled. Defaults to false.

---

**SFXTrack PlayerData::FootBubblesSound**

Sound to play when walking in water and coverage equals 1.0 (fully underwater).

---

**SFXTrack PlayerData::FootHardSound**

Sound to play when walking on a surface with Material footstepSoundId 1.

---

**SFXTrack PlayerData::FootMetalSound**

Sound to play when walking on a surface with Material footstepSoundId 2.

---

**ParticleEmitterData PlayerData::footPuffEmitter**

Particle emitter used to generate footpuffs (particles created as the
player walks along the ground).

**Note:**

The generation of foot puffs requires the appropriate triggers to be defined in the player's animation sequences. Without these, no foot puffs will be generated.

<table>
<thead>
<tr>
<th>int PlayerData::footPuffNumParts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of footpuff particles to generate each step.</td>
</tr>
</tbody>
</table>

Each foot puff is randomly placed within the defined foot puff radius. This includes having footPuffNumParts set to one.

**See also:**

footPuffRadius

<table>
<thead>
<tr>
<th>float PlayerData::footPuffRadius</th>
</tr>
</thead>
<tbody>
<tr>
<td>Particle creation radius for footpuff particles.</td>
</tr>
</tbody>
</table>

This is applied to each foot puff particle, even if footPuffNumParts is set to one. So set this value to zero if you want a single foot puff placed at exactly the same location under the player each time.

<table>
<thead>
<tr>
<th>SFXTrack PlayerData::FootShallowSound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sound to play when walking in water and coverage is less than footSplashHeight.</td>
</tr>
</tbody>
</table>

**See also:**

footSplashHeight
**SFXTrack PlayerData::FootSnowSound**

Sound to play when walking on a surface with `Material footstepSoundId 3`.

**SFXTrack PlayerData::FootSoftSound**

Sound to play when walking on a surface with `Material footstepSoundId 0`.

**float PlayerData::footstepSplashHeight**

Water coverage level to choose between `FootShallowSound` and `FootWadingSound`.

**See also:**

- `FootShallowSound`
- `FootWadingSound`

**SFXTrack PlayerData::FootUnderwaterSound**

Sound to play when walking in water and coverage equals 1.0 (fully underwater).

**SFXTrack PlayerData::FootWadingSound**

Sound to play when walking in water and coverage is less than 1, but > `footSplashHeight`.

**See also:**

- `footSplashHeight`
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>float PlayerData::groundImpactMinSpeed</code></td>
<td>Minimum falling impact speed to apply damage and initiate the camera shaking effect.</td>
</tr>
<tr>
<td><code>Point3F PlayerData::groundImpactShakeAmp</code></td>
<td>Amplitude of the camera shake effect after falling.</td>
</tr>
<tr>
<td></td>
<td>This is how much to shake the camera.</td>
</tr>
<tr>
<td><code>float PlayerData::groundImpactShakeDuration</code></td>
<td>Duration (in seconds) of the camera shake effect after falling.</td>
</tr>
<tr>
<td></td>
<td>This is how long to shake the camera.</td>
</tr>
<tr>
<td><code>float PlayerData::groundImpactShakeFalloff</code></td>
<td>Falloff factor of the camera shake effect after falling.</td>
</tr>
<tr>
<td></td>
<td>This is how to fade the camera shake over the duration.</td>
</tr>
<tr>
<td><code>Point3F PlayerData::groundImpactShakeFreq</code></td>
<td>Frequency of the camera shake effect after falling.</td>
</tr>
<tr>
<td></td>
<td>This is how fast to shake the camera.</td>
</tr>
<tr>
<td><code>float PlayerData::hardSplashSoundVelocity</code></td>
<td></td>
</tr>
</tbody>
</table>
Minimum velocity when entering the water for choosing between the impactWaterMedium and impactWaterHard sound to play.

See also:
- impactWaterMedium
- impactWaterHard

`float PlayerData::horizMaxSpeed`

Maximum horizontal speed.

Note:
This limit is only enforced if the player's horizontal speed exceeds `horizResistSpeed`.

See also:
- `horizResistSpeed`
- `horizResistFactor`

`float PlayerData::horizResistFactor`

Factor of resistance once `horizResistSpeed` has been reached.

See also:
- `horizMaxSpeed`
- `horizResistSpeed`

`float PlayerData::horizResistSpeed`

Horizontal speed at which resistance will take place.

See also:
<table>
<thead>
<tr>
<th><strong>horizMaxSpeed</strong></th>
<th><strong>horizResistFactor</strong></th>
</tr>
</thead>
</table>

**caseString PlayerData::imageAnimPrefix**

Optional prefix to all mounted image animation sequences in third person.

This defines a prefix that will be added when looking up mounted image animation sequences while in third person. It allows for the customization of a third person image based on the type of player.

**caseString PlayerData::imageAnimPrefixFP**

Optional prefix to all mounted image animation sequences in first person.

This defines a prefix that will be added when looking up mounted image animation sequences while in first person. It allows for the customization of a first person image based on the type of player.

**SFXTrack PlayerData::impactHardSound**

Sound to play after falling on a surface with Material footstepSoundId 1.

**SFXTrack PlayerData::impactMetalSound**

Sound to play after falling on a surface with Material footstepSoundId 2.

**SFXTrack PlayerData::impactSnowSound**
Sound to play after falling on a surface with Material\nfootstepSoundId 3.

**SFXTrack PlayerData::impactSoftSound**

Sound to play after falling on a surface with Material\nfootstepSoundId 0.

**SFXTrack PlayerData::impactWaterEasy**

Sound to play when entering the water with velocity <\nmediumSplashSoundVelocity.

**See also:**
   mediumSplashSoundVelocity

**SFXTrack PlayerData::impactWaterHard**

Sound to play when entering the water with velocity >=\nhardSplashSoundVelocity.

**See also:**
   hardSplashSoundVelocity

**SFXTrack PlayerData::impactWaterMedium**

Sound to play when entering the water with velocity >=\nmediumSplashSoundVelocity and < hardSplashSoundVelocity.

**See also:**
   mediumSplashSoundVelocity\nhardSplashSoundVelocity
float PlayerData::jetJumpEnergyDrain

Energy level drained each time the player jet jumps.

**Note:**
Setting this to zero will disable any energy drain

**See also:**
jetMinJumpEnergy

float PlayerData::jetJumpForce

Force used to accelerate the player when a jet jump is initiated.

float PlayerData::jetJumpSurfaceAngle

Angle from vertical (in degrees) where the player can jet jump.

float PlayerData::jetMaxJumpSpeed

Maximum vertical speed before the player can no longer jet jump.

float PlayerData::jetMinJumpEnergy

Minimum energy level required to jet jump.

**See also:**
jetJumpEnergyDrain

float PlayerData::jetMinJumpSpeed
Minimum speed needed to jet jump.

If the player's own z velocity is greater than this, then it is used to scale the jet jump speed, up to jetMaxJumpSpeed.

**See also:**

jetMaxJumpSpeed

```
int PlayerData::jumpDelay
```

Delay time in number of ticks ticks between jumps.

```
float PlayerData::jumpEnergyDrain
```

Energy level drained each time the player jumps.

**Note:**

Setting this to zero will disable any energy drain

**See also:**

minJumpEnergy

```
float PlayerData::jumpForce
```

Force used to accelerate the player when a jump is initiated.

```
float PlayerData::jumpSurfaceAngle
```

Angle from vertical (in degrees) where the player can jump.
**bool PlayerData::jumpTowardsNormal**

Controls the direction of the jump impulse.

When false, jumps are always in the vertical (+Z) direction. When true jumps are in the direction of the ground normal so long as the player is not directly facing the surface. If the player is directly facing the surface, then they will jump straight up.

**float PlayerData::landSequenceTime**

Time of land sequence play back when using new recover system.

If greater than 0 then the legacy fall recovery system will be bypassed in favour of just playing the player's land sequence. The time to recover from a fall then becomes this parameter's time and the land sequence's playback will be scaled to match.

**See also:**

* transitionToLand

**float PlayerData::maxBackwardSpeed**

Maximum backward speed when running.

**float PlayerData::maxCrouchBackwardSpeed**

Maximum backward speed when crouching.

**float PlayerData::maxCrouchForwardSpeed**

Maximum forward speed when crouching.
**float PlayerData::maxCrouchSideSpeed**

Maximum sideways speed when crouching.

**float PlayerData::maxForwardSpeed**

Maximum forward speed when running.

**float PlayerData::maxFreelookAngle**

Defines the maximum left and right angles (in radians) the player can look in freelook mode.

**float PlayerData::maxJumpSpeed**

Maximum vertical speed before the player can no longer jump.

**float PlayerData::maxLookAngle**

Highest angle (in radians) the player can look.

**Note:**

An angle of zero is straight ahead, with positive up and negative down.

**float PlayerData::maxProneBackwardSpeed**

Maximum backward speed when prone (laying down).
float PlayerData::maxProneForwardSpeed

Maximum forward speed when prone (laying down).

float PlayerData::maxProneSideSpeed

Maximum sideways speed when prone (laying down).

float PlayerData::maxSideSpeed

Maximum sideways speed when running.

float PlayerData::maxSprintBackwardSpeed

Maximum backward speed when sprinting.

float PlayerData::maxSprintForwardSpeed

Maximum forward speed when sprinting.

float PlayerData::maxSprintSideSpeed

Maximum sideways speed when sprinting.

float PlayerData::maxStepHeight

Maximum height the player can step up.

The player will automatically step onto changes in ground height less than maxStepHeight. The player will collide with ground
height changes greater than this.

**float PlayerData::maxTimeScale**

Maximum time scale for action animations.

If an action animation has a defined ground frame, it is automatically scaled to match the player's ground velocity. This field limits the maximum time scale used even if the player's velocity exceeds it.

**float PlayerData::maxUnderwaterBackwardSpeed**

Maximum backward speed when underwater.

**float PlayerData::maxUnderwaterForwardSpeed**

Maximum forward speed when underwater.

**float PlayerData::maxUnderwaterSideSpeed**

Maximum sideways speed when underwater.

**float PlayerData::mediumSplashSoundVelocity**

Minimum velocity when entering the water for choosing between the impactWaterEasy and impactWaterMedium sounds to play.

**See also:**

impactWaterEasy
impactWaterMedium
**float PlayerData::minImpactSpeed**

Minimum impact speed to apply falling damage.

This field also sets the minimum speed for the onImpact callback to be invoked.

**See also:**

ShapeBaseData::onImpact()

**float PlayerData::minJumpEnergy**

Minimum energy level required to jump.

**See also:**

jumpEnergyDrain

**float PlayerData::minJumpSpeed**

Minimum speed needed to jump.

If the player's own z velocity is greater than this, then it is used to scale the jump speed, up to maxJumpSpeed.

**See also:**

maxJumpSpeed

**float PlayerData::minLaterallImpactSpeed**

Minimum impact speed to apply non-falling damage.

This field also sets the minimum speed for the onLaterallImpact callback to be invoked.
See also:
  ShapeBaseData::onLaterallImpact()

float PlayerData::minLookAngle

Lowest angle (in radians) the player can look.

Note:
  An angle of zero is straight ahead, with positive up and negative down.

float PlayerData::minRunEnergy

Minimum energy level required to run or swim.

See also:
  runEnergyDrain

float PlayerData::minSprintEnergy

Minimum energy level required to sprint.

See also:
  sprintEnergyDrain

SFXTrack PlayerData::movingBubblesSound

Sound to play when in water and coverage equals 1.0 (fully underwater).

Note that unlike FootUnderwaterSound, this sound plays even if the player is not moving around in the water.
<table>
<thead>
<tr>
<th><strong>string PlayerData::physicsPlayerType</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Specifies the type of physics used by the player.</td>
</tr>
<tr>
<td>This depends on the physics module used. An example is 'Capsule'.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
</tr>
<tr>
<td>Not current used.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float PlayerData::pickupRadius</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Radius around the player to collide with Items in the scene (on server).</td>
</tr>
<tr>
<td>Internally the pickupRadius is added to the larger side of the initial bounding box to determine the actual distance, to a maximum of 2 times the bounding box size. The initial bounding box is that used for the root pose, and therefore doesn't take into account the change in pose.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Point3F PlayerData::proneBoundingBox</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Collision bounding box used when the player is prone (laying down).</td>
</tr>
<tr>
<td><strong>See also:</strong></td>
</tr>
<tr>
<td>boundingBox</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float PlayerData::proneForce</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Force used to accelerate the player when prone (laying down).</td>
</tr>
</tbody>
</table>
**int PlayerData::recoverDelay**

Number of ticks for the player to recover from falling.

**float PlayerData::recoverRunForceScale**

Scale factor applied to runForce while in the recover state.

This can be used to temporarily slow the player's movement after a fall, or prevent the player from moving at all if set to zero.

**bool PlayerData::renderFirstPerson**

Flag controlling whether to render the player shape in first person view.

**float PlayerData::runEnergyDrain**

Energy value drained each tick that the player is moving.

The player will not be able to move when his energy falls below minRunEnergy.

**Note:**

Setting this to zero will disable any energy drain.

**See also:**

minRunEnergy

**float PlayerData::runForce**
Force used to accelerate the player when running.

<table>
<thead>
<tr>
<th>Build</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>float</td>
<td><code>PlayerData::runSurfaceAngle</code> Maximum angle from vertical (in degrees) the player can run up.</td>
</tr>
<tr>
<td></td>
<td>filename</td>
<td><code>PlayerData::shapeNameFP[4]</code> File name of this player's shape that will be used in conjunction with the corresponding mounted image. These optional parameters correspond to each mounted image slot to indicate a shape that is rendered in addition to the mounted image shape. Typically these are a player's arms (or arm) that is animated along with the mounted image's state animation sequences.</td>
</tr>
<tr>
<td></td>
<td>SplashData</td>
<td><code>PlayerData::Splash</code> SplashData datablock used to create splashes when the player moves through water.</td>
</tr>
<tr>
<td></td>
<td>float</td>
<td><code>PlayerData::splashAngle</code> Maximum angle (in degrees) from pure vertical movement in water to generate splashes.</td>
</tr>
<tr>
<td></td>
<td>ParticleEmitterData</td>
<td><code>PlayerData::splashEmitter[3]</code> Particle emitters used to generate splash particles.</td>
</tr>
</tbody>
</table>
**float** PlayerData::splashFreqMod

Multiplied by speed to determine the number of splash particles to generate.

**float** PlayerData::splashVelEpsilon

Minimum speed to generate splash particles.

**float** PlayerData::splashVelocity

Minimum velocity when moving through water to generate splashes.

**bool** PlayerData::sprintCanJump

Can the player jump while sprinting.

**float** PlayerData::sprintEnergyDrain

Energy value drained each tick that the player is sprinting.

The player will not be able to move when his energy falls below sprintEnergyDrain.

**Note:**

Setting this to zero will disable any energy drain.

**See also:**

minSprintEnergy
**float PlayerData::sprintForce**

Force used to accelerate the player when sprinting.

**float PlayerData::sprintPitchScale**

Amount to scale pitch motion while sprinting.

**float PlayerData::sprintStrafeScale**

Amount to scale strafing motion vector while sprinting.

**float PlayerData::sprintYawScale**

Amount to scale yaw motion while sprinting.

**Point3F PlayerData::swimBoundingBox**

Collision bounding box used when the player is swimming.

**See also:**

- boundingBox

**float PlayerData::swimForce**

Force used to accelerate the player when swimming.

**bool PlayerData::transitionToLand**
When going from a fall to a land, should we transition between the two.

**Note:**

Only takes affect when landSequenceTime is greater than 0.

**See also:**

`landSequenceTime`

---

```cpp
float PlayerData::upMaxSpeed
```

Maximum upwards speed.

**Note:**

This limit is only enforced if the player's upward speed exceeds `upResistSpeed`.

**See also:**

`upResistSpeed`

`upResistFactor`

---

```cpp
float PlayerData::upResistFactor
```

Factor of resistance once `upResistSpeed` has been reached.

**See also:**

`upMaxSpeed`

`upResistSpeed`

---

```cpp
float PlayerData::upResistSpeed
```

Upwards speed at which resistance will take place.
See also:

- upMaxSpeed
- upResistFactor

SFXTrack PlayerData::waterBreathSound

Sound to play when in water and coverage equals 1.0 (fully underwater).

Note that unlike FootUnderwaterSound, this sound plays even if the player is not moving around in the water.
PointLight Class Reference
[Lighting]

Lighting object that radiates light in all directions. More...

Inheritance diagram for PointLight:

List of all members.
Public Attributes

Light

<table>
<thead>
<tr>
<th>float</th>
<th>radius</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Controls the falloff of the light emission.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Lighting object that radiates light in all directions.

*PointLight* is one of the two types of lighting objects that can be added to a Torque 3D level, the other being *SpotLight*. Unlike directional or conical light, the *PointLight* emits lighting in all directions. The attenuation is controlled by a single variable: `LightObject::radius`.

**Example:**

```plaintext
// Declaration of a point light in script,
new PointLight(CrystalLight) {

  radius = "10";
  isEnabled = "1";
  color = "1 0.905882 0 1";
  brightness = "0.5";
  castShadows = "1";
  priority = "1";
  animate = "1";
  animationType = "SubtlePulseLightAnim";
  animationPeriod = "3";
  animationPhase = "3";
  flareScale = "1";
  attenuationRatio = "0 1 1";
  shadowType = "DualParaboloidSinglePass"
  texSize = "512";
  overDarkFactor = "2000 1000 500 100"
  shadowDistance = "400";
  shadowSoftness = "0.15";
  numSplits = "1";
  logWeight = "0.91";
  fadeStartDistance = "0";
}
```
`lastSplitTerrainOnly = "0";`  
`splitFadeDistances = "10 20 30 40";`  
`representedInLightmap = "0";`  
`shadowDarkenColor = "0 0 0 -1";`  
`includeLightmappedGeometryInShadow = "1";`  
`position = "-61.3866 1.69186 5.1464";`  
`rotation = "1 0 0 0";`  

See also:  
- `LightBase`  
- `SpotLight`
Member Data Documentation

float PointLight::radius

Controls the falloff of the light emission.
Portal Class Reference
[Miscellaneous]

An object that provides a "window" into a zone, allowing a viewer to see what's rendered in the zone. More...

Inheritance diagram for Portal:

SimObject

NetObject

SceneObject

Zone

Portal

[legend]

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>bool</th>
<th>isExteriorPortal ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Test whether the portal connects interior zones to the outdoor zone.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>isInteriorPortal ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Test whether the portal connects interior zones only.</td>
</tr>
</tbody>
</table>
Public Attributes

Zoning

bool backSidePassable
   Whether one can view through the back-side of the portal.

bool frontSidePassable
   Whether one can view through the front-side of the portal.
Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Enables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Enables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
**Detailed Description**

An object that provides a "window" into a zone, allowing a viewer to see what's rendered in the zone.

A portal is an object that connects zones such that the content of one zone becomes visible in the other when looking through the portal.

Each portal is a full zone which is divided into two sides by the portal plane that intersects it. This intersection polygon is shown in red in the editor. Either of the sides of a portal can be connected to one or more zones.

A connection from a specific portal side to a zone is made in either of two ways:

1. By moving a Zone object to intersect with the portal at the respective side. While usually it makes sense for this overlap to be small, the connection is established correctly as long as the center of the Zone object that should connect is on the correct side of the portal plane.
2. By the respective side of the portal free of Zone objects that would connect to it. In this case, given that the other side is connected to one or more Zones, the portal will automatically connect itself to the outdoor "zone" which implicitly is present in any level.

From this, it follows that there are two types of portals:

**Exterior Portals**

An exterior portal is one that is connected to one or more Zone objects on one side and to the outdoor zone at the other side. This kind of portal is most useful for covering transitions from outdoor spaces to indoor spaces.

**Interior Portals**

An interior portal is one that is connected to one or more Zone objects on both sides. This kind of portal is most useful for covering transitions between indoor spaces.
Strictly speaking, there is a third type of portal called an "invalid portal". This is a portal that is not connected to a Zone object on either side in which case the portal serves no use.

Portals in Torque are bidirectional meaning that they connect zones both ways and you can look through the portal's front side as well as through its back-side.

Like Zones, Portals can either be box-shaped or use custom convex polyhedral shapes.

Portals will usually be created in the editor but can, of course, also be created in script code as such:

**Example:**

```cpp
// Example declaration of a Portal. This will create a box-shaped portal.
new Portal( PortalToTestZone )
{
    position = "12.8467 -4.02246 14.8017";
    rotation = "0 0 -1 97.5085";
    scale = "1 0.25 1";
    canSave = "1";
    canSaveDynamicFields = "1";
};
```

**Note:**
Keep in mind that zones and portals are more or less strictly a scene optimization mechanism meant to improve render times.

**See also:**
Zone
Member Function Documentation

```cpp
bool Portal::isExteriorPortal()
```

Test whether the portal connects interior zones to the outdoor zone.

**Returns:**
True if the portal is an exterior portal.

```cpp
bool Portal::isInteriorPortal()
```

Test whether the portal connects interior zones only.

**Returns:**
True if the portal is an interior portal.
Member Data Documentation

bool Portal::backSidePassable

Whether one can view through the back-side of the portal.

bool Portal::frontSidePassable

Whether one can view through the front-side of the portal.
PostEffect Class Reference
[Rendering]

A fullscreen shader effect. More...

Inheritance diagram for PostEffect:

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>void clearShaderMacros ()</strong></td>
<td>Remove all shader macros.</td>
</tr>
<tr>
<td><strong>void disable ()</strong></td>
<td>Disables the effect.</td>
</tr>
<tr>
<td><strong>String dumpShaderDisassembly ()</strong></td>
<td>Dumps this PostEffect shader's disassembly to a temporary text file.</td>
</tr>
<tr>
<td><strong>void enable ()</strong></td>
<td>Enables the effect.</td>
</tr>
<tr>
<td><strong>float getAspectRatio ()</strong></td>
<td></td>
</tr>
<tr>
<td><strong>bool isEnabled ()</strong></td>
<td></td>
</tr>
<tr>
<td><strong>void reload ()</strong></td>
<td>Reloads the effect shader and textures.</td>
</tr>
<tr>
<td><strong>void removeShaderMacro (string key)</strong></td>
<td>Remove a shader macro. This will usually be called within the preProcess callback.</td>
</tr>
<tr>
<td><strong>void setShaderConst (string name, string value)</strong></td>
<td>Sets the value of a uniform defined in the shader. This will usually be called within the setShaderConsts callback. Array type constants are not supported.</td>
</tr>
<tr>
<td><strong>void setShaderMacro (string key, string value=&quot;&quot;)</strong></td>
<td>Adds a macro to the effect's shader or sets an existing one's value. This will usually be called within the onAdd or preProcess callback.</td>
</tr>
<tr>
<td><strong>void setTexture (int index, string filePath)</strong></td>
<td>This is used to set the texture file and load the texture on a running effect. If the texture file is not different from the current file nothing is changed. If the texture cannot be found a null texture is assigned.</td>
</tr>
<tr>
<td><strong>bool toggle ()</strong></td>
<td></td>
</tr>
</tbody>
</table>
Toggles the effect between enabled / disabled.

**Callbacks**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>onAdd</code> ()</td>
<td>Called when this object is first created and registered.</td>
</tr>
<tr>
<td><code>onDisabled</code> ()</td>
<td>Called when this effect becomes disabled.</td>
</tr>
<tr>
<td><code>onEnabled</code> ()</td>
<td>Called when this effect becomes enabled. If the user returns false from this callback the effect will not be enabled.</td>
</tr>
<tr>
<td><code>preProcess</code> ()</td>
<td>Called when an effect is processed but before textures are bound. This allows the user to change texture related parameters or macros at runtime.</td>
</tr>
<tr>
<td><code>setShaderConsts</code> ()</td>
<td>Called immediate before processing this effect. This is the user's chance to set the value of shader uniforms (constants).</td>
</tr>
</tbody>
</table>
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>allowReflectPass</code></td>
<td>Is this effect processed during reflection render passes.</td>
</tr>
<tr>
<td>bool</td>
<td><code>isEnabled</code></td>
<td>Is the effect on.</td>
</tr>
<tr>
<td>bool</td>
<td><code>oneFrameOnly</code></td>
<td>Allows you to turn on a PostEffect for only a single frame.</td>
</tr>
<tr>
<td>bool</td>
<td><code>onThisFrame</code></td>
<td>Allows you to turn on a PostEffect for only a single frame.</td>
</tr>
<tr>
<td>string</td>
<td><code>renderBin</code></td>
<td>Name of a renderBin, used if renderTime is PFXBeforeBin or PFXAfterBin.</td>
</tr>
<tr>
<td>float</td>
<td><code>renderPriority</code></td>
<td>PostEffects are processed in DESCENDING order of renderPriority if more than one has the same renderBin/Time.</td>
</tr>
<tr>
<td>string</td>
<td><code>PFXRenderTime</code></td>
<td>When to process this effect during the frame.</td>
</tr>
<tr>
<td>string</td>
<td><code>shader</code></td>
<td>Name of a GFXShaderData for this effect.</td>
</tr>
<tr>
<td>bool</td>
<td><code>skip</code></td>
<td>Skip processing of this PostEffect and its children even if its parent is enabled. Parent and sibling PostEffects in the chain are still processed.</td>
</tr>
<tr>
<td>string</td>
<td><code>GFXStateBlockData</code></td>
<td>Name of a GFXStateBlockData for this effect.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>string <strong>target</strong></td>
<td>String identifier of this effect's target texture.</td>
<td></td>
</tr>
<tr>
<td><strong>PFXTargetClear</strong></td>
<td><strong>targetClear</strong> Describes when the target texture should be cleared.</td>
<td></td>
</tr>
<tr>
<td>ColorF <strong>targetClearColor</strong></td>
<td>Color to which the target texture is cleared before rendering.</td>
<td></td>
</tr>
<tr>
<td>string <strong>targetDepthStencil</strong></td>
<td>Optional string identifier for this effect's target depth/stencil texture.</td>
<td></td>
</tr>
<tr>
<td><strong>GFXFormat</strong></td>
<td><strong>targetFormat</strong> Format of the target texture, not applicable if writing to the backbuffer.</td>
<td></td>
</tr>
<tr>
<td>Point2F <strong>targetScale</strong></td>
<td>If targetSize is zero this is used to set a relative size from the current target.</td>
<td></td>
</tr>
<tr>
<td>Point2I <strong>targetSize</strong></td>
<td>If non-zero this is used as the absolute target size.</td>
<td></td>
</tr>
<tr>
<td><strong>filename</strong></td>
<td><strong>texture [6]</strong> Input textures to this effect (samplers).</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

A fullscreen shader effect.
PFXTextureIdentifiers
Member Function Documentation

void PostEffect::clearShaderMacros()

Remove all shader macros.

void PostEffect::disable()

Disables the effect.

String PostEffect::dumpShaderDisassembly()

Dumps this PostEffect shader's disassembly to a temporary text file.

Returns:
Full path to the dumped file or an empty string if failed.

void PostEffect::enable()

Enables the effect.

float PostEffect::getAspectRatio()

Returns:
Width over height of the backbuffer.

bool PostEffect::isEnabled()

Returns:
True if the effect is enabled.

```cpp
void PostEffect::onAdd() {
}
```

Called when this object is first created and registered.

```cpp
void PostEffect::onDisabled() {
}
```

Called when this effect becomes disabled.

```cpp
bool PostEffect::onEnabled() {
    Called when this effect becomes enabled. If the user returns false from this callback the effect will not be enabled.

    **Returns:**
    True to allow this effect to be enabled.

```cpp
void PostEffect::preProcess() {
    Called when an effect is processed but before textures are bound. This allows the user to change texture related parameters or macros at runtime.

    **Example:**
    ```cpp
    function SSAOPostFx::preProcess( %this ) {
        if ( $SSAOPostFx::quality != %this.quality ) {
            %this.quality = mClamp( mRound( $SSAOPostFx::quality ), 0, 2 );
    ```
    ```cpp
```
%this.setShaderMacro( "QUALITY", %this.quality );
%this.targetScale = $SSAOPostFx::targetScale;

See also:
setShaderConst
setShaderMacro

void PostEffect::reload( )

Reloads the effect shader and textures.

void PostEffect::removeShaderMacro(string key )

Remove a shader macro. This will usually be called within the preProcess callback.

Parameters:
    key  Macro to remove.

void PostEffect::setShaderConst(string name, string value )

Sets the value of a uniform defined in the shader. This will usually be called within the setShaderConsts callback. Array type constants are not supported.

Parameters:
    name  Name of the constant, prefixed with '$'.
    value Value to set, space separate values with more than one element.
Example:

```cpp
function MyPfx::setShaderConsts( %this )
{
    // example float4 uniform
    %this.setShaderConst( "$colorMod", "1.0
    // example float1 uniform
    %this.setShaderConst( "$strength", "3.0
    // example integer uniform
    %this.setShaderConst( "$loops", "5" );
}
```

```cpp
void PostEffect::setShaderConsts( )

Called immediate before processing this effect. This is the user's chance to set the value of shader uniforms (constants).

**See also:**

setShaderConst

```cpp
void PostEffect::setShaderMacro(string key,
    string value = ""
)

```

Adds a macro to the effect's shader or sets an existing one's value. This will usually be called within the onAdd or preProcess callback.

**Parameters:**

- `key` lval of the macro.
- `value` rval of the macro, or may be empty.

**Example:**
function MyPfx::onAdd( %this )
{
    %this.setShaderMacro( "NUM_SAMPLES", "%this.setShaderMacro( "HIGH_QUALITY_MODE"

    // In the shader looks like...
    // #define NUM_SAMPLES 10
    // #define HIGH_QUALITY_MODE
}

void PostEffect::setTexture(int index, string filePath)

This is used to set the texture file and load the texture on a running effect. If the texture file is not different from the current file nothing is changed. If the texture cannot be found a null texture is assigned.

Parameters:

index The texture stage index.

filePath The file name of the texture to set.

bool PostEffect::toggle( )

Toggles the effect between enabled / disabled.

Returns:

True if effect is enabled.
Member Data Documentation

**bool PostEffect::allowReflectPass**

Is this effect processed during reflection render passes.

**bool PostEffect::isEnabled**

Is the effect on.

**bool PostEffect::oneFrameOnly**

Allows you to turn on a PostEffect for only a single frame.

**bool PostEffect::onThisFrame**

Allows you to turn on a PostEffect for only a single frame.

**string PostEffect::renderBin**

Name of a renderBin, used if renderTime is PFXBeforeBin or PFXAfterBin.

**float PostEffect::renderPriority**

PostEffects are processed in DESCENDING order of renderPriority if more than one has the same renderBin/Time.

**PFXRenderTime PostEffect::renderTime**
When to process this effect during the frame.

<table>
<thead>
<tr>
<th><strong>string PostEffect::shader</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of a GFXShaderData for this effect.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>bool PostEffect::skip</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Skip processing of this PostEffect and its children even if its parent is enabled. Parent and sibling PostEffects in the chain are still processed.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>GFXStateBlockData PostEffect::stateBlock</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of a GFXStateBlockData for this effect.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>string PostEffect::target</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>String identifier of this effect's target texture.</td>
</tr>
</tbody>
</table>

**See also:**
- PFXTextureIdentifiers

<table>
<thead>
<tr>
<th><strong>PFXTargetClear PostEffect::targetClear</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Describes when the target texture should be cleared.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>ColorF PostEffect::targetClearColor</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Color to which the target texture is cleared before rendering.</td>
</tr>
<tr>
<td>string PostEffect::targetDepthStencil</td>
</tr>
<tr>
<td>---------------------------------------</td>
</tr>
<tr>
<td>Optional string identifier for this effect's target depth/stencil texture.</td>
</tr>
</tbody>
</table>

**See also:**

PFXTextureIdentifiers

<table>
<thead>
<tr>
<th>GFXFormat PostEffect::targetFormat</th>
</tr>
</thead>
<tbody>
<tr>
<td>Format of the target texture, not applicable if writing to the backbuffer.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Point2F PostEffect::targetScale</th>
</tr>
</thead>
<tbody>
<tr>
<td>If targetSize is zero this is used to set a relative size from the current target.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Point2I PostEffect::targetSize</th>
</tr>
</thead>
<tbody>
<tr>
<td>If non-zero this is used as the absolute target size.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>filename PostEffect::texture[6]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input textures to this effect ( samplers ).</td>
</tr>
</tbody>
</table>

**See also:**
PFXTextureIdentifiers
Precipitation Class Reference
[Special Effects, Atmosphere]

Defines a precipitation based storm (rain, snow, etc). More...

Inheritance diagram for Precipitation:

```
  SimObject
    ↓
  NetObject
    ↓
SceneObject
    ↓
  GameBase
    ↓
Precipitation
```

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void <strong>modifyStorm</strong> (float percentage=1.0f, float seconds=5.0f)</td>
<td>Smoothly change the maximum number of drops in the effect (from current value to numDrops * percentage).</td>
</tr>
<tr>
<td>void <strong>setPercentage</strong> (float percentage=1.0f)</td>
<td>Sets the maximum number of drops in the effect, as a percentage of numDrops.</td>
</tr>
<tr>
<td>void <strong>setTurbulence</strong> (float max=1.0f, float speed=5.0f, float seconds=5.0)</td>
<td>Smoothly change the turbulence parameters over a period of time.</td>
</tr>
</tbody>
</table>
## Public Attributes

### Rendering

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>animateSplashes</td>
<td>Set to true to enable splash animations when drops collide with other surfaces.</td>
</tr>
<tr>
<td>int</td>
<td>dropAnimateMS</td>
<td>Length (in milliseconds) to display each drop frame.</td>
</tr>
<tr>
<td>float</td>
<td>dropSize</td>
<td>Size of each drop of precipitation. This will scale the texture.</td>
</tr>
<tr>
<td>float</td>
<td>fadeDist</td>
<td>The distance at which drops begin to fade out.</td>
</tr>
<tr>
<td>float</td>
<td>fadeDistEnd</td>
<td>The distance at which drops are completely faded out.</td>
</tr>
<tr>
<td>ColorF</td>
<td>glowIntensity</td>
<td>Set to 0 to disable the glow or or use it to control the intensity of each channel.</td>
</tr>
<tr>
<td>bool</td>
<td>reflect</td>
<td>This enables precipitation rendering during reflection passes.</td>
</tr>
<tr>
<td>bool</td>
<td>rotateWithCamVel</td>
<td>Set to true to include the camera velocity when calculating drop rotation speed.</td>
</tr>
<tr>
<td>int</td>
<td>splashMS</td>
<td>Lifetime of splashes in milliseconds.</td>
</tr>
<tr>
<td>float</td>
<td>splashSize</td>
<td>Size of each splash animation when a drop collides with another surface.</td>
</tr>
<tr>
<td>bool</td>
<td>useLighting</td>
<td></td>
</tr>
</tbody>
</table>
Set to true to enable shading of the drops and splashes by the sun color.

<table>
<thead>
<tr>
<th>bool</th>
<th>useTrueBillboards</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Set to true to make drops true (non axis-aligned) billboards.</td>
</tr>
</tbody>
</table>

**Precipitation**

<table>
<thead>
<tr>
<th>float</th>
<th>boxHeight</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Height (vertical dimension) of the precipitation box.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>boxWidth</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Width and depth (horizontal dimensions) of the precipitation box.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int</th>
<th>numDrops</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maximum number of drops allowed to exist in the precipitation box at any one time.</td>
</tr>
</tbody>
</table>

**Collision**

<table>
<thead>
<tr>
<th>bool</th>
<th>doCollision</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Allow drops to collide with world objects.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>hitPlayers</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Allow drops to collide with Player objects; only valid if doCollision is true.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>hitVehicles</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Allow drops to collide with Vehicle objects; only valid if doCollision is true.</td>
</tr>
</tbody>
</table>

**Movement**

<table>
<thead>
<tr>
<th>bool</th>
<th>followCam</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Controls whether the Precipitation system follows the camera or remains where it is first placed in the scene.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>maxMass</th>
</tr>
</thead>
</table>
### Maximum

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>maxSpeed</td>
<td>Maximum speed at which a drop will fall.</td>
</tr>
</tbody>
</table>

### Minimum

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>minSpeed</td>
<td>Minimum speed at which a drop will fall.</td>
</tr>
</tbody>
</table>

### Boolean

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>useWind</td>
<td>Controls whether drops are affected by wind.</td>
</tr>
</tbody>
</table>

### Turbulence

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>maxTurbulence</td>
<td>Radius at which precipitation drops spiral when turbulence is enabled.</td>
</tr>
<tr>
<td>float</td>
<td>turbulenceSpeed</td>
<td>Speed at which precipitation drops spiral when turbulence is enabled.</td>
</tr>
<tr>
<td>bool</td>
<td>useTurbulence</td>
<td>Check to enable turbulence. This causes precipitation drops to spiral while falling.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td><strong>isRenderable</strong></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td><strong>isSelectable</strong></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Defines a precipitation based storm (rain, snow, etc).

The Precipitation effect works by creating many 'drops' within a fixed size box. This box can be configured to move around with the camera (to simulate level-wide precipitation), or to remain in a fixed position (to simulate localized precipitation). When followCam is true, the box containing the droplets can be thought of as centered on the camera then pushed slightly forward in the direction the camera is facing so most of the box is in front of the camera (allowing more drops to be visible on screen at once).

The effect can also be configured to create a small 'splash' whenever a drop hits another world object.

Example:

```cpp
// The following is added to a level file
new Precipitation( TheRain )
{
    dropSize = "0.5";
    splashSize = "0.5";
    splashMS = "250";
    animateSplashes = "1";
    dropAnimateMS = "0";
    fadeDist = "0";
    fadeDistEnd = "0";
    useTrueBillboards = "0";
    useLighting = "0";
    glowIntensity = "0 0 0 0";
    reflect = "0";
    rotateWithCamVel = "1";
    doCollision = "1";
    hitPlayers = "0";
    hitVehicles = "0";
}```
followCam = "1";
useWind = "0";
minSpeed = "1.5";
maxSpeed = "2";
minMass = "0.75";
maxMass = "0.85";
useTurbulence = "0";
maxTurbulence = "0.1";
turbulenceSpeed = "0.2";
numDrops = "1024";
boxWidth = "200";
boxHeight = "100";
dataBlock = "HeavyRain";

See also:

PrecipitationData
Member Function Documentation

```cpp
void Precipitation::modifyStorm (float percentage = 1.0f, float seconds = 5.0f)
```

Smoothly change the maximum number of drops in the effect (from current value to `numDrops * percentage`).

This method can be used to simulate a storm building or fading in intensity as the number of drops in the `Precipitation` box changes.

**Parameters:**

- `percentage` New maximum number of drops value (as a percentage of `numDrops`). Valid range is 0-1.
- `seconds` Length of time (in seconds) over which to increase the drops percentage value. Set to 0 to change instantly.

**Example:**

```cpp
%percentage = 0.5;   // The percentage, from 0 to 1
%seconds = 5.0;     // The length of time
%precipitation.modifyStorm( %percentage, %seconds );
```

```cpp
void Precipitation::setPercentage (float percentage = 1.0f)
```

Sets the maximum number of drops in the effect, as a percentage of `numDrops`.

The change occurs instantly (use `modifyStorm()` to change the number of drops over a period of time.

**Parameters:**

- New maximum number of drops value (as a
*percentage* percentage of *numDrops*. Valid range is 0-1.

Example:

```cpp
%percentage = 0.5; // The percentage, from 0 to 1, of the maximum drops to display
%precipitation.setPercentage(%percentage)
```

See also:

`modifyStorm`

```cpp
void Precipitation::setTurbulence(float max = 1.0f, float speed = 5.0f, float seconds = 5.0f)
```

Smoothly change the turbulence parameters over a period of time.

**Parameters:**

- `max` New *maxTurbulence* value. Set to 0 to disable turbulence.
- `speed` New *turbulenceSpeed* value.
- `seconds` Length of time (in seconds) over which to interpolate the turbulence settings. Set to 0 to change instantly.

Example:

```cpp
%turbulence = 0.5; // Set the new turbulence value.
%speed = 5.0; // The new speed of the turbulence effect.
%seconds = 5.0; // The length of time over which to make the change.
%precipitation.setTurbulence(%turbulence, %speed, %seconds)
```
Member Data Documentation

`bool Precipitation::animateSplashes`

Set to true to enable splash animations when drops collide with other surfaces.

`float Precipitation::boxHeight`

Height (vertical dimension) of the precipitation box.

`float Precipitation::boxWidth`

Width and depth (horizontal dimensions) of the precipitation box.

`bool Precipitation::doCollision`

Allow drops to collide with world objects.

If `animateSplashes` is true, drops that collide with another object will produce a simple splash animation.

**Note:**

This can be expensive as each drop will perform a raycast when it is created to determine where it will hit.

`int Precipitation::dropAnimateMS`

Length (in milliseconds) to display each drop frame.

If `dropAnimateMS` <= 0, drops select a single random frame at creation that does not change throughout the drop's lifetime. If
dropAnimateMS > 0, each drop cycles through the the available frames in the drop texture at the given rate.

float Precipitation::dropSize

Size of each drop of precipitation. This will scale the texture.

float Precipitation::fadeDist

The distance at which drops begin to fade out.

float Precipitation::fadeDistEnd

The distance at which drops are completely faded out.

bool Precipitation::followCam

Controls whether the Precipitation system follows the camera or remains where it is first placed in the scene.

Set to true to make it seem like it is raining everywhere in the level (ie. the Player will always be in the rain). Set to false to have a single area affected by rain (ie. the Player can move in and out of the rainy area).

ColorF Precipitation::glowIntensity

Set to 0 to disable the glow or or use it to control the intensity of each channel.

bool Precipitation::hitPlayers
Allow drops to collide with Player objects; only valid if doCollision is true.

**bool Precipitation::hitVehicles**

Allow drops to collide with Vehicle objects; only valid if doCollision is true.

**float Precipitation::maxMass**

Maximum mass of a drop.

Drop mass determines how strongly the drop is affected by wind and turbulence. On creation, the drop will be assigned a random speed between minMass and minMass.

**float Precipitation::maxSpeed**

Maximum speed at which a drop will fall.

On creation, the drop will be assigned a random speed between minSpeed and maxSpeed.

**float Precipitation::maxTurbulence**

Radius at which precipitation drops spiral when turbulence is enabled.

**float Precipitation::minMass**
Minimum mass of a drop.

Drop mass determines how strongly the drop is affected by wind and turbulence. On creation, the drop will be assigned a random speed between \textit{minMass} and \textit{minMass}.

\begin{verbatim}
float Precipitation::minSpeed
\end{verbatim}

Minimum speed at which a drop will fall.

On creation, the drop will be assigned a random speed between \textit{minSpeed} and \textit{maxSpeed}.

\begin{verbatim}
int Precipitation::numDrops
\end{verbatim}

Maximum number of drops allowed to exist in the precipitation box at any one time.

The actual number of drops in the effect depends on the current percentage, which can change over time using \textit{modifyStorm()}.

\begin{verbatim}
bool Precipitation::reflect
\end{verbatim}

This enables precipitation rendering during reflection passes.

**Note:**

This is expensive.

\begin{verbatim}
bool Precipitation::rotateWithCamVel
\end{verbatim}

Set to true to include the camera velocity when calculating drop rotation speed.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int Precipitation::splashMS</td>
<td>Lifetime of splashes in milliseconds.</td>
</tr>
<tr>
<td>float Precipitation::splashSize</td>
<td>Size of each splash animation when a drop collides with another surface.</td>
</tr>
<tr>
<td>float Precipitation::turbulenceSpeed</td>
<td>Speed at which precipitation drops spiral when turbulence is enabled.</td>
</tr>
<tr>
<td>bool Precipitation::useLighting</td>
<td>Set to true to enable shading of the drops and splashes by the sun color.</td>
</tr>
<tr>
<td>bool Precipitation::useTrueBillboards</td>
<td>Set to true to make drops true (non axis-aligned) billboards.</td>
</tr>
<tr>
<td>bool Precipitation::useTurbulence</td>
<td>Check to enable turbulence. This causes precipitation drops to spiral while falling.</td>
</tr>
</tbody>
</table>
bool Precipitation::useWind

Controls whether drops are affected by wind.

See also:
   ForestWindEmitter

Copyright © GarageGames, LLC. All Rights Reserved.
PrecipitationData Class Reference
[Special Effects, Atmosphere]

Defines the droplets used in a storm (raindrops, snowflakes, etc).

More...

Inheritance diagram for PrecipitationData:

List of all members.
### Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>dropShader</td>
<td>The name of the shader used for raindrops.</td>
</tr>
<tr>
<td>int</td>
<td>dropsPerSide</td>
<td>How many rows and columns are in the raindrop texture.</td>
</tr>
<tr>
<td>filename</td>
<td>dropTexture</td>
<td>Texture filename for drop particles.</td>
</tr>
<tr>
<td>SFXTrack</td>
<td>soundProfile</td>
<td>Looping SFXProfile effect to play while Precipitation is active.</td>
</tr>
<tr>
<td>int</td>
<td>splashesPerSide</td>
<td>How many rows and columns are in the splash texture.</td>
</tr>
<tr>
<td>string</td>
<td>splashShader</td>
<td>The name of the shader used for splashes.</td>
</tr>
<tr>
<td>filename</td>
<td>splashTexture</td>
<td>Texture filename for splash particles.</td>
</tr>
</tbody>
</table>
Detailed Description

Defines the droplets used in a storm (raindrops, snowflakes, etc).

Example:

```plaintext
datablock PrecipitationData( HeavyRain )
{
    soundProfile = "HeavyRainSound";
    dropTexture = "art/environment/precipitation/rain";
    splashTexture = "art/environment/precipitation/water_splash";
    dropsPerSide = 4;
    splashesPerSide = 2;
};
```

See also:

Precipitation
### Member Data Documentation

<table>
<thead>
<tr>
<th>Member Type</th>
<th>Name &amp; Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>PrecipitationData::dropShader</td>
<td>The name of the shader used for raindrops.</td>
</tr>
<tr>
<td>int</td>
<td>PrecipitationData::dropsPerSide</td>
<td>How many rows and columns are in the raindrop texture. For example, if the texture has 16 raindrops arranged in a grid, this field should be set to 4.</td>
</tr>
<tr>
<td>filename</td>
<td>PrecipitationData::dropTexture</td>
<td>Texture filename for drop particles. The drop texture can contain several different drop sub-textures arranged in a grid. There must be the same number of rows as columns. A random frame will be chosen for each drop.</td>
</tr>
<tr>
<td>SFXTrack</td>
<td>PrecipitationData::soundProfile</td>
<td>Looping SFXProfile effect to play while Precipitation is active.</td>
</tr>
<tr>
<td>int</td>
<td>PrecipitationData::splashesPerSide</td>
<td>How many rows and columns are in the splash texture. For example, if the texture has 9 splashes arranged in a grid, this field should be set to 3.</td>
</tr>
<tr>
<td>string PrecipitationData:splashShader</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The name of the shader used for splashes.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>filename PrecipitationData:splashTexture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Texture filename for splash particles.</td>
</tr>
<tr>
<td>The splash texture can contain several different splash sub-textures arranged in a grid. There must be the same number of rows as columns. A random frame will be chosen for each splash.</td>
</tr>
</tbody>
</table>
Prefab Class Reference
[Miscellaneous]

A collection of arbitrary objects which can be allocated and manipulated as a group. More...

Inheritance diagram for Prefab:

![Inheritance Diagram]

List of all members.
Public Member Functions

Callbacks

```c
void onLoad (SimGroup children)
    Called when the prefab file is loaded and children objects are created.
```
## Public Attributes

### Prefab

<p>| filename   | fileName | (.prefab) File describing objects within this prefab. |</p>
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td><strong>isRenderable</strong></td>
</tr>
<tr>
<td></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td><strong>isSelectable</strong></td>
</tr>
<tr>
<td></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

A collection of arbitrary objects which can be allocated and manipulated as a group.

Prefab always points to a (.prefab) file which defines its objects. In fact more than one Prefab can reference this file and both will update if the file is modified.

Prefab is a very simple object and only exists on the server. When it is created it allocates children objects by reading the (.prefab) file like a list of instructions. It then sets their transform relative to the Prefab and Torque networking handles the rest by ghosting the new objects to clients. Prefab itself is not ghosted.
Member Function Documentation

void Prefab::onLoad(SimGroup children)

Called when the prefab file is loaded and children objects are created.

Parameters:

children SimGroup containing all children objects.
Member Data Documentation

<table>
<thead>
<tr>
<th>filename Prefab::fileName</th>
</tr>
</thead>
</table>

(.prefab) File describing objects within this prefab.

Copyright © GarageGames, LLC. All Rights Reserved.
Base projectile class. Uses the ProjectileData class for properties of individual projectiles. More...

Inheritance diagram for Projectile:

List of all members.
Public Member Functions

void presimulate (float seconds=1.0f)
    Updates the projectile's positional and collision information.
# Public Attributes

## Physics

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point3F</td>
<td>initialPosition</td>
<td>Starting position for the projectile.</td>
</tr>
<tr>
<td>Point3F</td>
<td>initialVelocity</td>
<td>Starting velocity for the projectile.</td>
</tr>
</tbody>
</table>

## Source

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>sourceObject</td>
<td>ID number of the object that fired the projectile.</td>
</tr>
<tr>
<td>int</td>
<td>sourceSlot</td>
<td>The sourceObject's weapon slot that the projectile originates from.</td>
</tr>
</tbody>
</table>
# Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Enables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Enables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Base projectile class. Uses the ProjectileData class for properties of individual projectiles.
void Projectile::presimulate(float seconds = 1.0f)

Updates the projectile's positional and collision information.

This function will first delete the projectile if it is a server object and is outside it's ProjectileData::lifetime. Also responsible for applying gravity, determining collisions, triggering explosions, emitting trail particles, and calculating bounces if necessary.

Parameters:

seconds Amount of time, in seconds since the simulation's start, to advance.

Example:

```cpp
// Tell the projectile to process a simulation event that has passed since the simulation began.
%seconds = 2.0;
%projectile.presimulate(%seconds);
```

Note:

This function is not called if the SimObject::hidden is true.
Member Data Documentation

**Point3F Projectile::initialPosition**

Starting position for the projectile.

**Point3F Projectile::initialVelocity**

Starting velocity for the projectile.

**int Projectile::sourceObject**

ID number of the object that fired the projectile.

**Note:**

If the projectile was fired by a WeaponImage, sourceObject will be the object that owns the WeaponImage. This is usually the player.

**int Projectile::sourceSlot**

The sourceObject's weapon slot that the projectile originates from.
ProjectileData Class Reference
[Game Objects]

Stores properties for an individual projectile type. More...

Inheritance diagram for ProjectileData:

```
  SimObject
  |
  V
SimDataBlock
  |
  V
GameBaseData
  |
  V
ProjectileData
```

List of all members.
Public Member Functions

Callbacks

<table>
<thead>
<tr>
<th>void</th>
<th>onCollision (Projectile proj, SceneObject col, float fade, Point3F pos, Point3F normal)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Called when a projectile collides with another object.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>onExplode (Projectile proj, Point3F pos, float fade)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Called when a projectile explodes.</td>
</tr>
</tbody>
</table>
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>armingDelay</td>
<td>Amount of time, in milliseconds, before the projectile will cause damage or explode on impact.</td>
</tr>
<tr>
<td>float</td>
<td>bounceElasticity</td>
<td>Influences post-bounce velocity of a projectile that does not explode on contact.</td>
</tr>
<tr>
<td>float</td>
<td>bounceFriction</td>
<td>Factor to reduce post-bounce velocity of a projectile that does not explode on contact.</td>
</tr>
<tr>
<td>DecalData</td>
<td>decal</td>
<td>Decal datablock used for decals placed at projectile explosion points.</td>
</tr>
<tr>
<td>ExplosionData</td>
<td>Explosion</td>
<td>Explosion datablock used when the projectile explodes outside of water.</td>
</tr>
<tr>
<td>int</td>
<td>fadeDelay</td>
<td>Amount of time, in milliseconds, before the projectile begins to fade out.</td>
</tr>
<tr>
<td>float</td>
<td>gravityMod</td>
<td>Scales the influence of gravity on the projectile.</td>
</tr>
<tr>
<td>float</td>
<td>impactForce</td>
<td></td>
</tr>
<tr>
<td>bool</td>
<td>isBallistic</td>
<td>Determines if the projectile should be affected by gravity and whether or not it bounces before exploding.</td>
</tr>
<tr>
<td>int</td>
<td>lifetime</td>
<td>Amount of time, in milliseconds, before the projectile is removed from the simulation.</td>
</tr>
<tr>
<td><strong>LightDescription</strong></td>
<td>lightDesc</td>
<td></td>
</tr>
<tr>
<td>----------------------------</td>
<td>--------------------------------</td>
<td></td>
</tr>
<tr>
<td>LightDescription datablock used for lights attached to the projectile.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong></th>
<th>muzzleVelocity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amount of velocity the projectile receives from the &quot;muzzle&quot; of the gun.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>ParticleEmitterData</strong></th>
<th>ParticleEmitter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Particle emitter datablock used to generate particles while the projectile is outside of water.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>ParticleEmitterData</strong></th>
<th>particleWaterEmitter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Particle emitter datablock used to generate particles while the projectile is submerged in water.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>filename</strong></th>
<th>projectileShapeName</th>
</tr>
</thead>
<tbody>
<tr>
<td>File path to the model of the projectile.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Point3F</strong></th>
<th>scale</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scale to apply to the projectile's size.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>SFXTrack</strong></th>
<th>sound</th>
</tr>
</thead>
<tbody>
<tr>
<td>SFXTrack datablock used to play sounds while in flight.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>SplashData</strong></th>
<th>Splash</th>
</tr>
</thead>
<tbody>
<tr>
<td>Splash datablock used to create splash effects as the projectile enters or leaves water.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong></th>
<th>velInheritFactor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amount of velocity the projectile receives from the source that created it.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>ExplosionData</strong></th>
<th>waterExplosion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explosion datablock used when the projectile explodes underwater.</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

Stores properties for an individual projectile type.

Example:

datablock ProjectileData (GrenadeLauncherProjectile) {
    projectileShapeName = "art/shapes/weapons\n    directDamage = 30;
    radiusDamage = 30;
    damageRadius = 5;
    areaImpulse = 2000;
    explosion = GrenadeLauncherExplosion;
    waterExplosion = GrenadeLauncherWaterExplosion;
    decal = ScorchRXDecal;
    splash = GrenadeSplash;
    particleEmitter = GrenadeProjSmokeTrailEmitter;
    particleWaterEmitter = GrenadeTrailWaterEmitter;
    muzzleVelocity = 30;
    velInheritFactor = 0.3;
    armingDelay = 2000;
    lifetime = 10000;
    fadeDelay = 4500;
    bounceElasticity = 0.4;
    bounceFriction = 0.3;
    isBallistic = true;
    gravityMod = 0.9;
    lightDesc = GrenadeLauncherLightDesc;
    damageType = "GrenadeDamage";
};
Member Function Documentation

```cpp
void ProjectileData::onCollision (Projectile proj,
                  SceneObject col,
                  float fade,
                  Point3F pos,
                  Point3F normal)
```

Called when a projectile collides with another object.
This function is only called on server objects.

**Parameters:**

- `proj` The projectile colliding with `SceneObject` col.
- `col` The `SceneObject` hit by the projectile.
- `fade` The current fadeValue of the projectile, affects its visibility.
- `pos` The position of the collision.
- `normal` The normal of the collision.

**See also:**

- `Projectile`

```cpp
void ProjectileData::onExplode (Projectile proj,
                  Point3F pos,
                  float fade)
```

Called when a projectile explodes.
This function is only called on server objects.

**Parameters:**

- `proj` The exploding projectile.
pos  The position of the explosion.

fade The current fadeValue of the projectile, affects its visibility.

See also:

Projectile
## Member Data Documentation

<table>
<thead>
<tr>
<th>Member Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| int         | ProjectileData::armingDelay | Amount of time, in milliseconds, before the projectile will cause damage or explode on impact. This value must be equal to or less than the projectile's lifetime.

**See also:**
- lifetime

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>ProjectileData::bounceElasticity</td>
<td>Influences post-bounce velocity of a projectile that does not explode on contact. Scales the velocity from a bounce after friction is taken into account. A value of 1.0 will leave it's velocity unchanged while values greater than 1.0 will increase it.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>ProjectileData::bounceFriction</td>
<td>Factor to reduce post-bounce velocity of a projectile that does not explode on contact. Reduces bounce velocity by this factor and a multiple of the tangent to impact. Used to simulate surface friction.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DecalData</td>
<td>ProjectileData::decal</td>
</tr>
</tbody>
</table>

Decal datablock used for decals placed at projectile explosion points.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ExplosionData ProjectileData::Explosion</strong></td>
<td>Explosion datablock used when the projectile explodes outside of water.</td>
</tr>
<tr>
<td><strong>int ProjectileData::fadeDelay</strong></td>
<td>Amount of time, in milliseconds, before the projectile begins to fade out. This value must be smaller than the projectile's lifetime to have an affect.</td>
</tr>
<tr>
<td><strong>float ProjectileData::gravityMod</strong></td>
<td>Scales the influence of gravity on the projectile. The larger this value is, the more that gravity will affect the projectile. A value of 1.0 will assume &quot;normal&quot; influence upon it. The magnitude of gravity is assumed to be 9.81 m/s/s. <strong>Note:</strong> ProjectileData::isBallistic must be true for this to have any affect.</td>
</tr>
<tr>
<td><strong>float ProjectileData::impactForce</strong></td>
<td></td>
</tr>
<tr>
<td><strong>bool ProjectileData::isBallistic</strong></td>
<td>Determines if the projectile should be affected by gravity and whether or not it bounces before exploding.</td>
</tr>
</tbody>
</table>
**`int ProjectileData::lifetime`**

Amount of time, in milliseconds, before the projectile is removed from the simulation.

Used with `fadeDelay` to determine the transparency of the projectile at a given time. A projectile may exist up to a maximum of 131040ms (or 4095 ticks) as defined by `Projectile::MaxLivingTicks` in the source code.

**See also:**
- `fadeDelay`

**`LightDescription ProjectileData::lightDesc`**

`LightDescription` datablock used for lights attached to the projectile.

**`float ProjectileData::muzzleVelocity`**

Amount of velocity the projectile recieves from the "muzzle" of the gun.

Used with `velInheritFactor` to determine the initial velocity of the projectile. This value is never modified by the engine.

**Note:**
- This value by default is not transmitted between the server and the client.

**See also:**
- `velInheritFactor`

**`ParticleEmitterData ProjectileData::ParticleEmitter`**
Particle emitter datablock used to generate particles while the projectile is outside of water.

**Note:**
If datablocks are defined for both `particleEmitter` and `particleWaterEmitter`, both effects will play as the projectile enters or leaves water.

**See also:**
`particleWaterEmitter`

---

`ParticleEmitterData ProjectileData::particleWaterEmitter`

Particle emitter datablock used to generate particles while the projectile is submerged in water.

**Note:**
If datablocks are defined for both `particleWaterEmitter` and `particleEmitter`, both effects will play as the projectile enters or leaves water.

**See also:**
`particleEmitter`

---

`filename ProjectileData::projectileShapeName`

File path to the model of the projectile.

---

`Point3F ProjectileData::scale`

Scale to apply to the projectile's size.
**Note:**

This is applied after `SceneObject::scale`

**SFXTrack ProjectileData::sound**

*SFXTrack* datablock used to play sounds while in flight.

**SplashData ProjectileData::Splash**

*Splash* datablock used to create splash effects as the projectile enters or leaves water.

**float ProjectileData::velInheritFactor**

Amount of velocity the projectile receives from the source that created it.

Use an amount between 0 and 1 for the best effect. This value is never modified by the engine.

**Note:**

This value by default is not transmitted between the server and the client.

**ExplosionData ProjectileData::waterExplosion**

*Explosion* datablock used when the projectile explodes underwater.
ProximityMine Class Reference
[Game Objects]

A simple proximity mine. More...

Inheritance diagram for ProximityMine:

List of all members.
Public Member Functions

void explode ()
    Manually cause the mine to explode.
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

A simple proximity mine.

Proximity mines can be deployed using the world editor or thrown by an in-game object. Once armed, any Player or Vehicle object that moves within the mine's trigger area will cause it to explode.

Internally, the ProximityMine object transitions through the following states:

1. **Thrown**: Mine has been thrown, but has not yet attached to a surface.
2. **Deployed**: Mine has attached to a surface but is not yet armed. Start playing the armingSound and armed sequence.
3. **Armed**: Mine is armed and will trigger if a Vehicle or Player object moves within the trigger area.
4. **Triggered**: Mine has been triggered and will explode soon. Invoke the onTriggered callback, and start playing the triggerSound and triggered sequence.
5. **Exploded**: Mine has exploded and will be deleted on the server shortly. Invoke the onExplode callback on the server and generate the explosion effects on the client.

Note:

Proximity mines with the static field set to true will start in the Armed state. Use this for mines placed with the World Editor.

The shape used for the mine may optionally define the following sequences:

**armed**

Sequence to play when the mine is deployed, but before it becomes active and triggerable (armingDelay should be set appropriately).

**triggered**

Sequence to play when the mine is triggered, just before it explodes (triggerDelay should be set appropriately).
Example:

datablock ProximityMineData( SimpleMine )
{
    // ShapeBaseData fields
    category = "Weapon";
    shapeFile = "art/shapes/weapons/misc/proximityMine.dts"

    // ItemData fields
    sticky = true;

    // ProximityMineData fields
    armingDelay = 0.5;
    armingSound = MineArmedSound;

    autoTriggerDelay = 0;
    triggerOnOwner = true;
    triggerRadius = 5.0;
    triggerSpeed = 1.0;
    triggerDelay = 0.5;
    triggerSound = MineTriggeredSound;
    explosion = RocketLauncherExplosion;

    // dynamic fields
    pickUpName = "Proximity Mines";
    maxInventory = 20;

    damageType = "MineDamage"; // type of damage
    radiusDamage = 30; // amount of damage
    damageRadius = 8; // search radius
    areaImpulse = 2000; // magnitude
};

function ProximityMineData::onTriggered( %

See also:

ProximityMineData
Member Function Documentation

```cpp
void ProximityMine::explode()
```

Manually cause the mine to explode.
ProximityMineData Class Reference
[Game Objects]

Stores common properties for a ProximityMine. More...

Inheritance diagram for ProximityMineData:

```
SimObject
  ↓
SimDataBlock
  ↓
GameBaseData
  ↓
ShapeBaseData
  ↓
ItemData
  ↓
ProximityMineData
```

List of all members.
Public Member Functions

Callbacks

```c
void onExplode (ProximityMine obj, Point3F pos)
    Callback invoked when a ProximityMine is about to explode.

void onTriggered (ProximityMine obj, SceneObject target)
    Callback invoked when an object triggers the ProximityMine.
```
## Public Attributes

### Arming

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>armingDelay</td>
<td>Delay (in seconds) from when the mine is placed to when it becomes active.</td>
</tr>
<tr>
<td>SFXTrack</td>
<td>armingSound</td>
<td>Sound to play when the mine is armed (starts at the same time as the <em>armed</em> sequence if defined).</td>
</tr>
</tbody>
</table>

### Triggering

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>autoTriggerDelay</td>
<td>Delay (in seconds) from arming until the mine automatically triggers and explodes, even if no object has entered the trigger area.</td>
</tr>
<tr>
<td>float</td>
<td>triggerDelay</td>
<td>Delay (in seconds) from when the mine is triggered until it explodes.</td>
</tr>
<tr>
<td>bool</td>
<td>triggerOnOwner</td>
<td>Controls whether the mine can be triggered by the object that owns it.</td>
</tr>
<tr>
<td>float</td>
<td>triggerRadius</td>
<td>Distance at which an activated mine will detect other objects and explode.</td>
</tr>
<tr>
<td>SFXTrack</td>
<td>triggerSound</td>
<td>Sound to play when the mine is triggered (starts at the same time as the <em>triggered</em> sequence if defined).</td>
</tr>
<tr>
<td>float</td>
<td>triggerSpeed</td>
<td>Speed above which moving objects within the trigger radius will trigger the mine.</td>
</tr>
</tbody>
</table>
**Explosion**

<table>
<thead>
<tr>
<th>float</th>
<th>explosionOffset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Offset from the mine's origin where the explosion emanates from. Sometimes a thrown mine may be slightly sunk into the ground. This can be just enough to cause the explosion to occur under the ground, especially on flat ground, which can end up blocking the explosion. This offset along the mine's 'up' normal allows you to raise the explosion origin to a better height.</td>
</tr>
</tbody>
</table>
Detailed Description

Stores common properties for a ProximityMine.

See also:

ProximityMine
Member Function Documentation

```cpp
void ProximityMineData::onExplode(ProximityMine obj, Point3F pos)
```

Callback invoked when a ProximityMine is about to explode.

**Parameters:**
- `obj` The ProximityMine object
- `pos` The position of the mine explosion

**Note:**
This callback is only invoked on the server.

**See also:**
ProximityMine

```cpp
void ProximityMineData::onTriggered(ProximityMine obj, SceneObject target)
```

Callback invoked when an object triggers the ProximityMine.

**Parameters:**
- `obj` The ProximityMine object
- `target` The object that triggered the mine

**Note:**
This callback is only invoked on the server.

**See also:**
ProximityMine
## Member Data Documentation

### float `ProximityMineData::armingDelay`

Delay (in seconds) from when the mine is placed to when it becomes active.

### SFXTrack `ProximityMineData::armingSound`

Sound to play when the mine is armed (starts at the same time as the *armed* sequence if defined).

### float `ProximityMineData::autoTriggerDelay`

Delay (in seconds) from arming until the mine automatically triggers and explodes, even if no object has entered the trigger area.

Set to 0 to disable.

### float `ProximityMineData::explosionOffset`

Offset from the mine's origin where the explosion emanates from. Sometimes a thrown mine may be slightly sunk into the ground. This can be just enough to cause the explosion to occur under the ground, especially on flat ground, which can end up blocking the explosion. This offset along the mine's 'up' normal allows you to raise the explosion origin to a better height.

### float `ProximityMineData::triggerDelay`

Delay (in seconds) from when the mine is triggered until it
explodes.

```cpp
bool ProximityMineData::triggerOnOwner
```

Controls whether the mine can be triggered by the object that owns it.

For example, a player could deploy mines that are only dangerous to other players and not himself.

```cpp
float ProximityMineData::triggerRadius
```

Distance at which an activated mine will detect other objects and explode.

```cpp
SFXTrack ProximityMineData::triggerSound
```

Sound to play when the mine is triggered (starts at the same time as the `triggered` sequence if defined).

```cpp
float ProximityMineData::triggerSpeed
```

Speed above which moving objects within the trigger radius will trigger the mine.
PxCloth Class Reference
[Physics]

Rectangular patch of cloth simulated by PhysX. More...

Inheritance diagram for PxCloth:

- SimObject
- NetObject
- SceneObject
- GameBase
- PxCloth

List of all members.
### Public Attributes

<table>
<thead>
<tr>
<th>PxClothAttachment</th>
<th>attachments</th>
<th>Optional way to specify cloth verts that will be attached to the world position it is created at.</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>bending</td>
<td>Enables or disables bending resistance.</td>
</tr>
<tr>
<td>float</td>
<td>bendingStiffness</td>
<td>Bending stiffness of the cloth in the range 0 to 1.</td>
</tr>
<tr>
<td>bool</td>
<td>damping</td>
<td>Enable/disable damping of internal velocities.</td>
</tr>
<tr>
<td>float</td>
<td>dampingCoefficient</td>
<td>Spring damping of the cloth in the range 0 to 1.</td>
</tr>
<tr>
<td>float</td>
<td>density</td>
<td>Density of the cloth (Mass per Area).</td>
</tr>
<tr>
<td>float</td>
<td>friction</td>
<td>Friction coefficient in the range 0 to 1.</td>
</tr>
<tr>
<td>string</td>
<td>Material</td>
<td>Name of the material to render.</td>
</tr>
<tr>
<td>Point2I</td>
<td>samples</td>
<td>The number of cloth vertices in width and length.</td>
</tr>
<tr>
<td>bool</td>
<td>selfCollision</td>
<td>Enables or disables self-collision handling within a single piece of cloth.</td>
</tr>
<tr>
<td>Point2F</td>
<td>size</td>
<td>The width and height of the cloth.</td>
</tr>
<tr>
<td>float</td>
<td>thickness</td>
<td>Value representing how thick the cloth is.</td>
</tr>
</tbody>
</table>
bool triangleCollision
Not supported in current release (according to PhysX docs).
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Rectangular patch of cloth simulated by PhysX.

PxCloth is affected by other objects in the simulation but does not itself affect others, it is essentially a visual effect. Eg, shooting at cloth will disturb it but will not explode the projectile.

Be careful with the cloth size and resolution because it can easily become performance intensive to simulate. A single piece of cloth that is very large or high resolution is also much more expensive than multiple pieces that add up to the same number of verts.

Note that most field docs have been copied from their PhysX counterpart.
## Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>PxCloth::attachment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optional way to specify cloth verts that will be attached to the world position it is created at.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>PxCloth::bending</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enables or disables bending resistance.</td>
<td></td>
</tr>
<tr>
<td>Set the bending resistance through PxCloth::bendingStiffness.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>PxCloth::bendingStiffness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bending stiffness of the cloth in the range 0 to 1.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>PxCloth::damping</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable/disable damping of internal velocities.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>PxCloth::dampingCoefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spring damping of the cloth in the range 0 to 1.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>PxCloth::density</th>
</tr>
</thead>
<tbody>
<tr>
<td>Density of the cloth (Mass per Area).</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Variable</td>
</tr>
<tr>
<td>----------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>float</td>
<td>PxCloth::friction</td>
</tr>
<tr>
<td>string</td>
<td>PxCloth::Material</td>
</tr>
<tr>
<td>Point2I</td>
<td>PxCloth::samples</td>
</tr>
<tr>
<td>bool</td>
<td>PxCloth::selfCollision</td>
</tr>
<tr>
<td>Point2F</td>
<td>PxCloth::size</td>
</tr>
<tr>
<td>float</td>
<td>PxCloth::thickness</td>
</tr>
</tbody>
</table>
the maximal distance between two adjacent cloth particles in their rest pose. Visual artifacts or collision problems may appear if the thickness is too small.

```cpp
bool PxCloth::triangleCollision
```

Not supported in current release (according to PhysX docs).

Enables or disables collision detection of cloth triangles against the scene. If not set, only collisions of cloth particles are detected. If set, collisions of cloth triangles are detected as well.
PxMaterial Class Reference
Physics

Defines a PhysX material assignable to a PxMaterial. More...

Inheritance diagram for PxMaterial:

List of all members.
**Public Attributes**

**PxMaterial**

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>dynamicFriction</td>
<td>Coefficient of dynamic friction to be applied.</td>
</tr>
<tr>
<td>float</td>
<td>restitution</td>
<td>Coefficient of a bounce applied to the shape in response to a collision.</td>
</tr>
<tr>
<td>float</td>
<td>staticFriction</td>
<td>Coefficient of static friction to be applied.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Defines a PhysX material assignable to a `PxMaterial`.

When two actors collide, the collision behavior that results depends on the material properties of the actors' surfaces. For example, the surface properties determine if the actors will or will not bounce, or if they will slide or stick. Currently, the only special feature supported by materials is anisotropic friction, but according to Nvidia, other effects such as moving surfaces and more types of friction are slotted for future release.

For more information, refer to Nvidia's PhysX docs.
Member Data Documentation

float PxBall::dynamicFriction

Coefficient of dynamic friction to be applied.

Dynamic friction reduces the velocity of a moving object while it is in contact with a surface. A higher coefficient will result in a larger reduction in velocity. A shape's dynamicFriction should be equal to or larger than 0.

float PxBall::restitution

Coefficient of a bounce applied to the shape in response to a collision.

A value of 0 makes the object bounce as little as possible, while higher values up to 1.0 result in more bounce.

Note:
Values close to or above 1.0 may cause stability problems and/or increasing energy.

float PxBall::staticFriction

Coefficient of static friction to be applied.

Static friction determines the force needed to start moving an at-rest object in contact with a surface. If the force applied onto shape cannot overcome the force of static friction, the shape will remain at rest. A higher coefficient will require a larger force to start motion.

Note:
This value should be larger than 0.
PxMultiActor Class Reference

[Physics]

Represents a destructible physical object simulated using PhysX.

More...

Inheritance diagram for PxMultiActor:

```
    SimObject
       ↑
      NetObject
         ↑
        SceneObject
           ↑
          GameBase
             ↑
            PxMultiActor
```

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void listMeshes</td>
<td>Lists all meshes of the provided type in the console window.</td>
</tr>
<tr>
<td>void setAllHidden</td>
<td>Sets the <code>PxMultiActor</code> to a broken or unbroken state.</td>
</tr>
<tr>
<td>void setBroken</td>
<td>Hides or unhides all meshes contained in the <code>PxMultiActor</code>.</td>
</tr>
<tr>
<td>void setMeshHidden</td>
<td>Prevents the provided mesh from being rendered.</td>
</tr>
</tbody>
</table>

```cpp
void listMeshes (enum Hidden/Shown/All)  
    Lists all meshes of the provided type in the console window.

void setAllHidden (bool)  
    Hides or unhides all meshes contained in the PxMultiActor.

void setBroken (bool)  
    Sets the PxMultiActor to a broken or unbroken state.

void setMeshHidden (string meshName, bool isHidden)  
    Prevents the provided mesh from being rendered.
```
### Public Attributes

#### Debug

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>broken</td>
</tr>
<tr>
<td>bool</td>
<td>debugRender</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Represents a destructible physical object simulated using PhysX.

Usually it is preferred to use PhysicsShape and not PxMultiActor because it is not PhysX specific and much easier to setup.

See also:

PxMultiActorData.
Member Function Documentation

void PxMultiActor::listMeshes(enum Hidden/Shown/ All )

Lists all meshes of the provided type in the console window.

Parameters:

    All          Lists all of the PxMultiActor's meshes.
    Hidden       Lists all of the PxMultiActor's hidden meshes.
    Shown        Lists all of the PxMultiActor's visible meshes.

void PxMultiActor::setAllHidden( bool )

Hides or unhides all meshes contained in the PxMultiActor.

Hidden meshes will not be rendered.

void PxMultiActor::setBroken( bool )

Sets the PxMultiActor to a broken or unbroken state.

void PxMultiActor::setMeshHidden( string meshName,  
                                 bool isHidden )

Prevents the provided mesh from being rendered.
**Member Data Documentation**

<table>
<thead>
<tr>
<th>boolean PxMultiActor::broken</th>
</tr>
</thead>
</table>

| boolean PxMultiActor::debugRender |

Copyright © GarageGames, LLC. All Rights Reserved.
PxMultiActorData Class Reference

[Physics]

Defines the properties of a type of PxA MultiActor. More...

Inheritance diagram for PxMultiActorData:

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>breakForce</td>
<td>Force required to break an actor.</td>
</tr>
<tr>
<td>bool</td>
<td>clientOnly</td>
<td></td>
</tr>
<tr>
<td>void</td>
<td>dumpModel</td>
<td>Dumps model hierarchy and details to a file.</td>
</tr>
<tr>
<td>void</td>
<td>reload</td>
<td>Reloads all data used for the PxMultiActorData.</td>
</tr>
<tr>
<td>bool</td>
<td>singlePlayerOnly</td>
<td></td>
</tr>
</tbody>
</table>

## Physics

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>angularDrag</td>
<td>Value used to help calculate rotational drag force while submerged in water.</td>
</tr>
<tr>
<td>float</td>
<td>buoyancyDensity</td>
<td>The density used to calculate buoyant forces.</td>
</tr>
<tr>
<td>float</td>
<td>linearDrag</td>
<td>Value used to help calculate linear drag force while submerged in water.</td>
</tr>
<tr>
<td>PxMaterial</td>
<td>Material</td>
<td>An optional PxMaterial to be used for the PxMultiActor.</td>
</tr>
<tr>
<td>bool</td>
<td>noCorrection</td>
<td></td>
</tr>
<tr>
<td>filename</td>
<td>physXStream</td>
<td>.XML file containing data such as actors, shapes, and joints.</td>
</tr>
<tr>
<td>string</td>
<td></td>
<td></td>
</tr>
<tr>
<td>float</td>
<td>waterDragScale</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Scale to apply to linear and angular dampening while submerged in water.</td>
<td></td>
</tr>
</tbody>
</table>

**Media**

<table>
<thead>
<tr>
<th>filename</th>
<th>shapeName</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Path to the .DAE or .DTS file to render.</td>
</tr>
</tbody>
</table>
Detailed Description

Defines the properties of a type of `PxMultiActor`.

Usually it is preferred to use `PhysicsShape` rather than `PxMultiActor` because a `PhysicsShape` is not PhysX specific and can be much easier to setup.

For more information, refer to Nvidia's PhysX docs.
Member Data Documentation

float PxMultiActorData::angularDrag

Value used to help calculate rotational drag force while submerged in water.

float PxMultiActorData::breakForce

Force required to break an actor.

This value does not apply to joints. If an actor is associated with a joint it will break whenever the joint does. This allows an actor "not" associated with a joint to also be breakable.

float PxMultiActorData::buoyancyDensity

The density used to calculate buoyant forces.

The result of the calculated buoyancy is relative to the density of the WaterObject the PxMultiActor is within.

Note:

This value is necessary because Torque 3D does its own buoyancy simulation. It is not handled by PhysX.

See also:

WaterObject::density

bool PxMultiActorData::clientOnly

void PxMultiActorData::dumpModel
Dumps model hierarchy and details to a file.

The file will be created as 'model.dump' in the game folder. If model.dump already exists, it will be overwritten.

**float PxMultiActorData::linearDrag**

Value used to help calculate linear drag force while submerged in water.

**PxMaterial PxMultiActorData::Material**

An optional PxMaterial to be used for the PxMultiActor.

Defines properties such as friction and restitution. Unrelated to the material used for rendering. The physXStream will contain defined materials that can be customized in 3DS Max. To override the material for all physics shapes in the physXStream, specify a material here.

**bool PxMultiActorData::noCorrection**

**filename PxMultiActorData::physXStream**

.XML file containing data such as actors, shapes, and joints.

These files can be created using a free PhysX plugin for 3DS Max.

**void PxMultiActorData::reload**

Reloads all data used for the PxMultiActorData.
If the reload successfully completes, all PxMultiActor's will be notified.

**filename** `PxMultiActorData::shapeName`

Path to the .DAE or .DTS file to render.

**bool** `PxMultiActorData::singlePlayerOnly`

**PxMultiActorData::string**

**float** `PxMultiActorData::waterDragScale`

Scale to apply to linear and angular dampening while submerged in water.
RadialImpulseEvent Class Reference

[Physics]

Creates a physics-based impulse effect from a defined central point and magnitude. More...

List of all members.
### Static Public Member Functions

<table>
<thead>
<tr>
<th>static void send (string inPosition=&quot;1.0 1.0 1.0&quot;, float radius=10.0f, float magnitude=20.0f)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Applies a radial impulse to any SceneObjects within the area of effect.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Creates a physics-based impulse effect from a defined central point and magnitude.

**See also:**

RadialImpulseEvent::send
Member Function Documentation

```cpp
static void RadialImpulseEvent::send (string inPosition = "1.0 1.0 1.0",
          float  radius = 10.0f,
          float  magnitude = 20.0f)
```

Applies a radial impulse to any SceneObjects within the area of effect.

This event is performed both server and client-side.

**Parameters:**

- `position`: Center point for this radial impulse.
- `radius`: Distance from the position for this radial impulse to affect.
- `magnitude`: The force applied to objects within the radius from the position of this radial impulse effect.

**Example:**

```cpp
// Define the Position
%position = "10.0 15.0 10.0";

// Define the Radius
%radius = "25.0";

// Define the Magnitude
%magnitude = "30.0"

// Create a globalRadialImpulse physics effect
RadialImpulseEvent::send(%position,%radius,%magnitude);
```
ReflectorDesc Class Reference

[Miscellaneous]

A datablock which defines performance and quality properties for dynamic reflections. More...

Inheritance diagram for ReflectorDesc:

![Inheritance diagram](legend)

List of all members.
# Public Attributes

## ReflectorDesc

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>detailAdjust</td>
<td>Scale applied to lod calculation of objects rendering into this reflection (modulates $pref::TS::detailAdjust$).</td>
</tr>
<tr>
<td>float</td>
<td>farDist</td>
<td>Far plane distance to use when rendering reflections.</td>
</tr>
<tr>
<td>int</td>
<td>maxRateMs</td>
<td>If less than maxRateMs has elapsed since this reflection was last updated, then do not update it again. This 'skip' can be disabled by setting maxRateMs to zero.</td>
</tr>
<tr>
<td>float</td>
<td>nearDist</td>
<td>Near plane distance to use when rendering this reflection. Adjust this to limit self-occlusion artifacts.</td>
</tr>
<tr>
<td>int</td>
<td>objectTypeMask</td>
<td>Object types which render into this reflection.</td>
</tr>
<tr>
<td>float</td>
<td>priority</td>
<td>Priority for updating this reflection, relative to others.</td>
</tr>
<tr>
<td>int</td>
<td>texSize</td>
<td>Size in pixels of the (square) reflection texture. For a cubemap this value is interpreted as size of each face.</td>
</tr>
<tr>
<td>bool</td>
<td>useOcclusionQuery</td>
<td>If available on the device use HOQs to determine if the reflective object is visible before updating its reflection.</td>
</tr>
</tbody>
</table>
Detailed Description

A datablock which defines performance and quality properties for dynamic reflections.

ReflectorDesc is not itself a reflection and does not render reflections. It is a dummy class for holding and exposing to the user a set of reflection related properties. Objects which support dynamic reflections may then reference a ReflectorDesc.

Example:

datablock ReflectorDesc( ExampleReflectorDesc ) {
    texSize = 256;
    nearDist = 0.1;
    farDist = 500;
    objectTypeMask = 0xFFFFFFFF;
    detailAdjust = 1.0;
    priority = 1.0;
    maxRateMs = 0;
    useOcclusionQuery = true;
};

See also:
    ShapeBaseData::cubeReflectorDesc
Member Data Documentation

float ReflectorDesc::detailAdjust

Scale applied to lod calculation of objects rendering into this reflection ( modulates $pref::TS::detailAdjust ).

float ReflectorDesc::farDist

Far plane distance to use when rendering reflections.

int ReflectorDesc::maxRateMs

If less than maxRateMs has elapsed since this reflection was last updated, then do not update it again. This 'skip' can be disabled by setting maxRateMs to zero.

float ReflectorDesc::nearDist

Near plane distance to use when rendering this reflection. Adjust this to limit self-occlusion artifacts.

int ReflectorDesc::objectTypeMask

Object types which render into this reflection.

float ReflectorDesc::priority

Priority for updating this reflection, relative to others.
**int ReflectorDesc::texSize**

Size in pixels of the (square) reflection texture. For a cubemap this value is interpreted as size of each face.

**bool ReflectorDesc::useOcclusionQuery**

If available on the device use HOQs to determine if the reflective object is visible before updating its reflection.
The abstract base for all render bins. More...

Inheritance diagram for RenderBinManager:

List of all members.
Public Member Functions

string getBinType ()
Returns the bin type string.
## Public Attributes

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>binType</td>
<td>Sets the render bin type which limits what render instances are added to this bin.</td>
</tr>
<tr>
<td>float</td>
<td>processAddOrder</td>
<td>Defines the order for adding instances in relation to other bins.</td>
</tr>
<tr>
<td>float</td>
<td>renderOrder</td>
<td>Defines the order for rendering in relation to other bins.</td>
</tr>
</tbody>
</table>
Detailed Description

The abstract base for all render bins.

The render bins are used by the engine as a high level method to order and batch rendering operations.
string RenderBinManager::getBinType()

Returns the bin type string.
## Member Data Documentation

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>RenderBinManager::binType</td>
<td>Sets the render bin type which limits what render instances are added to this bin.</td>
</tr>
<tr>
<td>float</td>
<td>RenderBinManager::processAddOrder</td>
<td>Defines the order for adding instances in relation to other bins.</td>
</tr>
<tr>
<td>float</td>
<td>RenderBinManager::renderOrder</td>
<td>Defines the order for rendering in relation to other bins.</td>
</tr>
</tbody>
</table>
**RenderFormatToken Class Reference**

[GFX]

Used to change the render target format when rendering in AL.

More...

Inheritance diagram for RenderFormatToken:

```
SimObject

RenderPassStateToken

RenderFormatToken [legend]
```

List of all members.
### Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><code>aaLevel</code></td>
<td>Anti-aliasing level for the this token. 0 disables, -1 uses adapter default.</td>
</tr>
<tr>
<td>PostEffect</td>
<td><code>copyEffect</code></td>
<td>This PostEffect will be run when the render target is changed to the format specified by this token. It is used to copy/format data into the token rendertarget.</td>
</tr>
<tr>
<td>GFXFormat</td>
<td><code>depthFormat</code></td>
<td>Sets the depth/stencil buffer format for this token.</td>
</tr>
<tr>
<td>GFXFormat</td>
<td><code>format</code></td>
<td>Sets the color buffer format for this token.</td>
</tr>
<tr>
<td>PostEffect</td>
<td><code>resolveEffect</code></td>
<td>This PostEffect will be run when the render target is changed back to the format active prior to this token. It is used to copy/format data from the token rendertarget to the backbuffer.</td>
</tr>
</tbody>
</table>
Detailed Description

Used to change the render target format when rendering in AL.

*RenderFormatToken* is an implementation which changes the format of the back buffer and/or the depth buffer.

The *RenderPassStateBin* manager changes the rendering state associated with this token. In stock Torque 3D, a single example exists in the way of *AL_FormatToken* (found in renderManager.cs). In that script file, all the render managers are initialized, and a single *RenderFormatToken* is used. This implementation basically exists to ensure Advanced Lighting works with MSAA.

The actions for this token toggle the format of the back/depth buffers and it lets you specify a custom shader to "copy" the data so it can be reformatted or altered. This is done through the variables *copyEffect* and *resolveEffect* (which are post processes just like fog or glow).

Example:

```csharp
// This token, and the associated render managers, ensure that driver MSAA does not get used for Advanced Lighting renders.
// The 'AL_FormatResolve' PostEffect copies the result to the backbuffer.
new RenderFormatToken(AL_FormatToken)
{
    enabled = "false";

    format = "GFXFormatR8G8B8A8";
    depthFormat = "GFXFormatD24S8";
    aaLevel = 0; // -1 = match backbuffer

    // The contents of the back buffer before
    // is provided in $inTex
    copyEffect = "AL_FormatCopy";

    // The contents of the render target created by this format token is
```
// provided in $inTex
resolveEffect = "AL_FormatCopy";
};

See also:

RenderPassToken
RenderPassStateBin
game/core/scripts/client/renderManager.cs
### Member Data Documentation

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>int RenderFormatToken::aaLevel</code></td>
<td>Anti-aliasing level for this token. 0 disables, -1 uses adapter default.</td>
</tr>
<tr>
<td><code>PostEffect RenderFormatToken::copyEffect</code></td>
<td>This PostEffect will be run when the render target is changed to the format specified by this token. It is used to copy/format data into the token render target.</td>
</tr>
<tr>
<td><code>GFXFormat RenderFormatToken::depthFormat</code></td>
<td>Sets the depth/stencil buffer format for this token.</td>
</tr>
<tr>
<td><code>GFXFormat RenderFormatToken::format</code></td>
<td>Sets the color buffer format for this token.</td>
</tr>
<tr>
<td><code>PostEffect RenderFormatToken::resolveEffect</code></td>
<td>This PostEffect will be run when the render target is changed back to the format active prior to this token. It is used to copy/format data from the token render target to the back buffer.</td>
</tr>
</tbody>
</table>
RenderGlowMgr Class Reference
[Render Binning]

A render bin for the glow pass. More...

Inheritance diagram for RenderGlowMgr:

List of all members.
**Detailed Description**

A render bin for the glow pass.

When the glow buffer PostEffect is enabled this bin gathers mesh render instances with glow materials and renders them to the glowbuffer offscreen render target.

This render target is then used by the 'GlowPostFx' PostEffect to blur and render the glowing portions of the screen.
RenderImposterMgr Class Reference
[Render Binning]

A render bin for batch rendering imposters. More...

Inheritance diagram for RenderImposterMgr:

List of all members.
Detailed Description

A render bin for batch rendering imposters.

This render bin gathers imposter render instances and renders them in large batches.

You can type 'metrics( imposter )' in the console to see rendering statistics.
RenderMeshExample Class Reference
[Examples]

An example scene object which renders a mesh. More...

Inheritance diagram for RenderMeshExample:

List of all members.
Public Member Functions

void postApply ()
    A utility method for forcing a network update.
Public Attributes

Rendering

<table>
<thead>
<tr>
<th>string</th>
<th>Material</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The name of the material used to render the mesh.</td>
</tr>
</tbody>
</table>
# Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
**Detailed Description**

An example scene object which renders a mesh.

This class implements a basic `SceneObject` that can exist in the world at a 3D position and render itself. There are several valid ways to render an object in Torque. This class implements the preferred rendering method which is to submit a `MeshRenderInst` along with a `Material`, vertex buffer, primitive buffer, and transform and allow the `RenderMeshMgr` handle the actual setup and rendering for you.

See the C++ code for implementation details.
Member Function Documentation

`void RenderMeshExample::postApply()`

A utility method for forcing a network update.
Member Data Documentation

<table>
<thead>
<tr>
<th>string RenderMeshExample::Material</th>
</tr>
</thead>
</table>

The name of the material used to render the mesh.

Copyright © GarageGames, LLC. All Rights Reserved.
RenderMeshMgr Class Reference
[Render Binning]

A render bin for mesh rendering. More...

Inheritance diagram for RenderMeshMgr:

- SimObject
- RenderBinManager
- RenderMeshMgr

List of all members.
Detailed Description

A render bin for mesh rendering.

This is the primary render bin in Torque which does most of the work of rendering DTS shapes and arbitrary mesh geometry. It knows how to render mesh instances using materials and supports hardware mesh instancing.
RenderObjectExample Class Reference

[Examples]

An example scene object which renders using a callback. More...

Inheritance diagram for RenderObjectExample:

List of all members.
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td><code>isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td><code>isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

An example scene object which renders using a callback.

This class implements a basic SceneObject that can exist in the world at a 3D position and render itself. Note that RenderObjectExample handles its own rendering by submitting itself as an ObjectRenderInst (see renderInstance enderPassmanager.h) along with a delegate for its render() function. However, the preferred rendering method in the engine is to submit a MeshRenderInst along with a Material, vertex buffer, primitive buffer, and transform and allow the RenderMeshMgr handle the actual rendering. You can see this implemented in RenderMeshExample.

See the C++ code for implementation details.

Copyright © GarageGames, LLC. All Rights Reserved.
RenderObjectMgr Class Reference
[Render Binning]

A render bin which uses object callbacks for rendering. More...

Inheritance diagram for RenderObjectMgr:

List of all members.
Detailed Description

A render bin which uses object callbacks for rendering.

This render bin gathers object render instances and calls its delegate method to perform rendering. It is used infrequently for specialized scene objects which perform custom rendering.
RenderOcclusionMgr Class Reference
[Render Binning]

A render bin which renders occlusion query requests. More...

Inheritance diagram for RenderOcclusionMgr:

List of all members.
### Static Public Attributes

<table>
<thead>
<tr>
<th>static bool</th>
<th>debugRender</th>
</tr>
</thead>
<tbody>
<tr>
<td>A debugging feature which renders the occlusion volumes to the scene.</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

A render bin which renders occlusion query requests.

This render bin gathers occlusion query render instances and renders them. It is currently used by light flares and ShapeBase reflection cubemaps.

You can type ‘$RenderOcclusionMgr::debugRender = true’ in the console to see debug rendering of the occlusion geometry.
RenderParticleMgr Class Reference
[Render Binning]

A render bin which renders particle geometry. More...

Inheritance diagram for RenderParticleMgr:

```
SimObject

<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RenderBinManager</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td>RenderTargetBinManager</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td>RenderParticleMgr</td>
</tr>
</tbody>
</table>
```

List of all members.
Detailed Description

A render bin which renders particle geometry.

This render bin gathers particle render instances, sorts, and renders them. It is currently used by ParticleEmitter and LightFlareData.
RenderPassManager Class Reference

[Render Binning]

A grouping of render bin managers which forms a render pass.

More...

Inheritance diagram for RenderPassManager:

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void <strong>addManager</strong> <em>(RenderBinManager renderBin)</em></td>
<td>Add as a render bin manager to the pass.</td>
</tr>
<tr>
<td>RenderBinManager <strong>getManager</strong> <em>(int index)</em></td>
<td>Returns the render bin manager at the index or null if the index is out of range.</td>
</tr>
<tr>
<td>int <strong>getManagerCount</strong> ()</td>
<td>Returns the total number of bin managers.</td>
</tr>
<tr>
<td>void <strong>removeManager</strong> <em>(RenderBinManager renderBin)</em></td>
<td>Removes a render bin manager.</td>
</tr>
</tbody>
</table>
Detailed Description

A grouping of render bin managers which forms a render pass.

The render pass is used to order a set of `RenderBinManager` objects which are used when rendering a scene. This class does little work itself other than managing its list of render bins.

In 'core/scripts/client/renderManager.cs' you will find the DiffuseRenderPassManager which is used by the C++ engine to render the scene.

See also:

- `RenderBinManager`
Member Function Documentation

```cpp
void RenderPassManager::addManager(RenderBinManager renderBin)

Add as a render bin manager to the pass.
```

```cpp
RenderBinManager RenderPassManager::getManager(int index)

Returns the render bin manager at the index or null if the index is out of range.
```

```cpp
int RenderPassManager::getManagerCount()

Returns the total number of bin managers.
```

```cpp
void RenderPassManager::removeManager(RenderBinManager renderBin)

Removes a render bin manager.
```
RenderPassStateBin Class Reference
[Render Binning]

A non-rendering render bin used to enable/disable a RenderPassStateToken. More...

Inheritance diagram for RenderPassStateBin:

- SimObject
  - RenderBinManager
    - RenderPassStateBin

List of all members.
Public Attributes

| RenderPassStateToken       | stateToken       |
Detailed Description

A non-rendering render bin used to enable/disable a RenderPassStateToken.

This is a utility RenderBinManager which does not render any render instances. Its only used to define a point in the render bin order at which a RenderPassStateToken is triggered.

See also:

RenderPassStateToken
## Member Data Documentation

| **RenderPassStateToken** | **RenderPassStateBin::stateToken** |

Copyright © GarageGames, LLC. All Rights Reserved.
RenderPassStateToken Class Reference
[Render Binning]

Abstract base class for RenderFormatToken, used to manipulate what goes on in the render manager. More...

Inheritance diagram for RenderPassStateToken:

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>void</th>
<th>disable ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables the token.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>enable ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Enables the token.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>toggle ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Toggles the token from enabled to disabled or vice versa.</td>
</tr>
</tbody>
</table>
### Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>enabled</td>
<td>Enables or disables this token.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Abstract base class for `RenderFormatToken`, used to manipulate what goes on in the render manager.

You cannot actually instantiate `RenderPassToken`, only its child: `RenderFormatToken`. `RenderFormatToken` is an implementation which changes the format of the back buffer and/or the depth buffer.

The `RenderPassStateBin` manager changes the rendering state associated with a token it is declared with. In stock Torque 3D, a single example exists in the way of `AL_FormatToken` (found in `renderManager.cs`). In that script file, all the render managers are intialized, and a single `RenderFormatToken` is used. This implementation basically exists to ensure Advanced Lighting works with MSAA.

**See also:**
- `RenderFormatToken`
- `RenderPassStateBin`
- `game/core/scripts/client/renderManager.cs`
## Member Function Documentation

<table>
<thead>
<tr>
<th>Function Definition</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void RenderPassStateToken::disable()</code></td>
<td>Disables the token.</td>
</tr>
<tr>
<td><code>void RenderPassStateToken::enable()</code></td>
<td>Enables the token.</td>
</tr>
<tr>
<td><code>void RenderPassStateToken::toggle()</code></td>
<td>Toggles the token from enabled to disabled or vice versa.</td>
</tr>
</tbody>
</table>
Member Data Documentation

`bool RenderPassStateToken::enabled`

Enables or disables this token.
RenderPrePassMgr Class Reference
[Render Binning]

The render bin which performs a z+normals prepass used in Advanced Lighting. More...

Inheritance diagram for RenderPrePassMgr:

List of all members.
Detailed Description

The render bin which performs a z+normals prepass used in Advanced Lighting.

This render bin is used in Advanced Lighting to gather all opaque mesh render instances and render them to the g-buffer for use in lighting the scene and doing effects.

PostEffect and other shaders can access the output of this bin by using the prepass texture target name. See the edge anti-aliasing post effect for an example.

See also:

    game/core/scripts/client/postFx/edgeAA.cs

Copyright © GarageGames, LLC. All Rights Reserved.
RenderShapeExample Class Reference

[Examples]

An example scene object which renders a DTS. More...

Inheritance diagram for RenderShapeExample:

```
SimObject
  
NetObject
  
SceneObject
  
RenderShapeExample
```

List of all members.
Public Attributes

Rendering

<table>
<thead>
<tr>
<th>filename</th>
<th>shapeFile</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The path to the DTS shape file.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Static bool</th>
<th>isRenderable</th>
<th>Disables rendering of all instances of this type.</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

An example scene object which renders a DTS.

This class implements a basic SceneObject that can exist in the world at a 3D position and render itself. There are several valid ways to render an object in Torque. This class makes use of the 'TS' (three space) shape system. TS manages loading the various mesh formats supported by Torque as well as rendering those meshes (including LOD and animation...though this example doesn't include any animation over time).

See the C++ code for implementation details.
Member Data Documentation

filename RenderShapeExample::shapeFile

The path to the DTS shape file.
RenderTerrainMgr Class Reference
[Render Binning]

A render bin for terrain mesh rendering. More...

Inheritance diagram for RenderTerrainMgr:

List of all members.
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>renderWireframe</td>
<td>Used to enable wireframe rendering on terrain for debugging.</td>
</tr>
</tbody>
</table>
Detailed Description

A render bin for terrain mesh rendering.

This bin renders terrain render instances from a **TerrainBlock**.
Normally a mesh would render via the **RenderMeshMgr**, but terrain uses a **TerrainMaterial** designed for multi-layered surfaces which this bin can processs.

Copyright © GarageGames, LLC. All Rights Reserved.
RenderTexTargetBinManager Class Reference

[Render Binning]

An abstract base class for render bin managers that render to a named texture target. More...

Inheritance diagram for RenderTexTargetBinManager:

List of all members.
Detailed Description

An abstract base class for render bin managers that render to a named texture target.

This bin itself doesn't do any rendering work. It offers functionality to manage a texture render target which derived render bin classes can render into.

See also:

RenderPrePassMgr
RenderTranslucentMgr Class Reference
[Render Binning]

A render bin for rendering translucent meshes. More...

Inheritance diagram for RenderTranslucentMgr:

List of all members.
Detailed Description

A render bin for rendering translucent meshes.

This bin is used to render translucent render mesh instances and render object instances. It is generally ordered late in the RenderPassManager after all opaque geometry bins.
The **RigidShape** class implements rigid-body physics for DTS objects in the world. More...

Inheritance diagram for RigidShape:

```
  SimObject
    NetObject
      SceneObject
        GameBase
          ShapeBase
            RigidShape
```

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>void</th>
<th>freezeSim (bool isFrozen)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Enables or disables the physics simulation on the RigidShape object.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>reset ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Clears physic forces from the shape and sets it at rest.</td>
</tr>
</tbody>
</table>

### Callbacks

<table>
<thead>
<tr>
<th>void</th>
<th>onEnterLiquid (string objId, string waterCoverage, string liquidType)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Called whenever this RigidShape object enters liquid.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>onLeaveLiquid (string objId, string liquidType)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Called whenever the RigidShape object exits liquid.</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td><code>isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td><code>isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

The **RigidShape** class implements rigid-body physics for DTS objects in the world.

"Rigid body physics" refers to a system whereby objects are assumed to have a finite size, equally distributed masses, and where deformations of the objects themselves are not accounted for. Uses the **RigidShape** class to control its physics.

Example:

```plaintext
datablock RigidShapeData( BouncingBoulder ) {

category = "RigidShape";

shapeFile = "~/data/shapes/boulder/boulder.dts";
emap = true;

// Rigid Body
mass = 500;
massCenter = "0 0 0";    // Center of mass
massBox = "0 0 0";      // Size of massBox
    // if zero it defaults to object bounding box

drag = 0.2;            // Drag coefficient
bodyFriction = 0.2;
bodyRestitution = 0.1;
minImpactSpeed = 5;    // Impact over this invoke the script callback
softImpactSpeed = 5;   // Play Soft Impact Sound
hardImpactSpeed = 15;  // Play Hard Impact Sound
integration = 4;       // Physics integration:
collisionTol = 0.1;    // Collision distance tolerance
contactTol = 0.1;      // Contact velocity tolerance

minRollSpeed = 10;
```

maxDrag = 0.5;
minDrag = 0.01;

dustHeight = 10;
dragForce = 0.05;
vertFactor = 0.05;
};

new RigidShape()
{
    dataBlock = "BouncingBoulder";
    parentGroup = EWCreatorWindow.objectGroup;
};

See also:
    RigidShapeData
    ShapeBase
void RigidShape::freezeSim(bool isFrozen )

Enables or disables the physics simulation on the RigidShape object.

Parameters:

    isFrozen  Boolean frozen state to set the object.

Example:

    // Define the frozen state.
    %isFrozen = "true";
    
    // Inform the object of the defined frozen state
    %thisRigidShape.freezeSim(%isFrozen);

See also:

    ShapeBaseData

void RigidShape::onEnterLiquid(string objId, string waterCoverage, string liquidType)

Called whenever this RigidShape object enters liquid.

Parameters:

    objId    The ID of the rigidShape object.
    waterCoverage    Amount of water coverage the RigidShape has.
    liquidType    Type of liquid that was entered.
Example:

```cpp
// The RigidShape object falls in a body of liquid, causing the callback to occur.
RigidShape::onEnterLiquid(%this,%objId,%waterCoverage,%liquidType)
{
    // Code to run whenever this callback occurs.
}
```

See also:

`ShapeBase`

```cpp
void RigidShape::onLeaveLiquid(string objId,
                                string liquidType)
```

Called whenever the `RigidShape` object exits liquid.

**Parameters:**

- `objId` The ID of the `RigidShape` object.
- `liquidType` Type of liquid that was exited.

Example:

```cpp
// The RigidShape object exits in a body of liquid, causing the callback to occur.
RigidShape::onLeaveLiquid(%this,%objId,%liquidType)
{
    // Code to run whenever this callback occurs.
}
```

See also:

`ShapeBase`

```cpp
void RigidShape::reset()
```
Clears physic forces from the shape and sets it at rest.

**Example:**

```cpp
// Inform the RigidShape object to reset.
%thisRigidShape.reset();
```

**See also:**

[ShapeBaseData](#)
RigidShapeData Class Reference

[Platform]

Defines the physics properties for an individual RigidShapeData physics object. More...

Inheritance diagram for RigidShapeData:

```
SimObject

SimDataBlock

GameBaseData

ShapeBaseData

RigidShapeData

[legend]
```

List of all members.
**Public Attributes**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>bodyFriction</code></td>
<td>How much friction this object has. Lower values will cause the object to appear to be more slippery.</td>
</tr>
<tr>
<td>float</td>
<td><code>bodyRestitution</code></td>
<td>The percentage of kinetic energy kept by this object in a collision.</td>
</tr>
<tr>
<td>float</td>
<td><code>collisionTol</code></td>
<td>Collision distance tolerance.</td>
</tr>
<tr>
<td>float</td>
<td><code>contactTol</code></td>
<td>Contact velocity tolerance.</td>
</tr>
<tr>
<td>float</td>
<td><code>hardImpactSpeed</code></td>
<td>Minimum speed at which the object must be travelling for the hard impact sound to be played.</td>
</tr>
<tr>
<td>int</td>
<td><code>integration</code></td>
<td>Number of physics steps to process per tick.</td>
</tr>
<tr>
<td>Point3F</td>
<td><code>massBox</code></td>
<td>Size of inertial box.</td>
</tr>
<tr>
<td>Point3F</td>
<td><code>massCenter</code></td>
<td>Center of mass for rigid body.</td>
</tr>
<tr>
<td>float</td>
<td><code>maxDrag</code></td>
<td>Maximum drag available to this object.</td>
</tr>
<tr>
<td>float</td>
<td><code>minDrag</code></td>
<td>Minimum drag available to this object.</td>
</tr>
<tr>
<td>float</td>
<td><code>minImpactSpeed</code></td>
<td>Minimum collision speed to classify collision as impact (triggers onImpact on server object).</td>
</tr>
<tr>
<td>float</td>
<td><code>minRollSpeed</code></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Variable</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>float</td>
<td>softImpactSpeed</td>
<td>Minimum speed at which this object must be travelling for the soft impact sound to be played.</td>
</tr>
</tbody>
</table>

**Camera**

*The settings used by the shape when it is the camera.*

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>cameraDecay</td>
<td>Scalar rate at which the third person camera offset decays, per tick.</td>
</tr>
<tr>
<td>float</td>
<td>cameraLag</td>
<td>Scalar amount by which the third person camera lags the object, relative to the object’s linear velocity.</td>
</tr>
<tr>
<td>float</td>
<td>cameraOffset</td>
<td>The vertical offset of the object’s camera.</td>
</tr>
<tr>
<td>bool</td>
<td>cameraRoll</td>
<td>Specifies whether the camera’s rotation matrix, and the render eye transform are multiplied during camera updates.</td>
</tr>
</tbody>
</table>

**Forces**

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>dragForce</td>
<td>Used to simulate the constant drag acting on the object.</td>
</tr>
<tr>
<td>float</td>
<td>vertFactor</td>
<td>The scalar applied to the vertical portion of the velocity drag acting on a object.</td>
</tr>
</tbody>
</table>

**Particle Effects**

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ParticleEmitterData</td>
<td>dustEmitter</td>
<td></td>
</tr>
</tbody>
</table>
Array of pointers to `ParticleEmitterData` datablocks which will be used to emit particles at object/terrain contact point.

<table>
<thead>
<tr>
<th>float</th>
<th><code>dustHeight</code></th>
<th>Height of dust effects.</th>
</tr>
</thead>
</table>

Particle emitter used to create a dust trail for the moving object.

<table>
<thead>
<tr>
<th><code>ParticleSystem</code></th>
<th><code>dustTrailEmitter</code></th>
<th>Array of pointers to <code>ParticleEmitterData</code> datablocks which will generate splash effects.</th>
</tr>
</thead>
</table>

The simulated frequency modulation of a splash generated by this object. Multiplied along with speed and time elapsed when determining splash emission rate.

<table>
<thead>
<tr>
<th>float</th>
<th><code>splashFreqMod</code></th>
<th>The threshold speed at which we consider the object's movement to have stopped when updating splash effects.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th><code>splashVelEpsilon</code></th>
<th>Maximum height from the ground at which the object will generate dust.</th>
</tr>
</thead>
</table>

**Sounds**

`SFXTrack` *exitingWater*

The AudioProfile will be used to produce sounds when emerging from water.

<table>
<thead>
<tr>
<th><code>SFXTrack</code></th>
<th><code>exitSplashSoundVelocity</code></th>
<th>The minimum velocity at which the exit splash sound will be played when emerging from water.</th>
</tr>
</thead>
</table>

`SFXTrack` *hardImpactSound*
<table>
<thead>
<tr>
<th>SFXTrack</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>impactWaterEasy</td>
<td>The AudioProfile will be used to produce sounds when a soft impact with water occurs.</td>
</tr>
<tr>
<td>impactWaterHard</td>
<td>The AudioProfile will be used to produce sounds when a hard impact with water occurs.</td>
</tr>
<tr>
<td>impactWaterMedium</td>
<td>The AudioProfile will be used to produce sounds when a medium impact with water occurs.</td>
</tr>
<tr>
<td>waterWakeSound</td>
<td>The AudioProfile will be used to produce sounds when a water wake is displayed.</td>
</tr>
</tbody>
</table>

- **Sound** to play when body impacts with at least **hardImpactSpeed**.

- **float** `hardSplashSoundVelocity`  
The minimum velocity at which the hard splash sound will be played when impacting water.

- **float** `mediumSplashSoundVelocity`  
The minimum velocity at which the medium splash sound will be played when impacting water.

- **float** `softSplashSoundVelocity`  
The minimum velocity at which the soft splash sound will be played when impacting water.
Detailed Description

Defines the physics properties for an individual RigidShapeData physics object.

Example:

datablock RigidShapeData( BouncingBoulder )
{
    category = "RigidShape";

    shapeFile = "~/data/shapes/boulder/boulder.dts";
    emap = true;

    // Rigid Body
    mass = 500;
    massCenter = "0 0 0";    // Center of mass
    massBox = "0 0 0";      // Size of box used for moment of inertia,
                           // if zero it defaults to object bounding box
    drag = 0.2;             // Drag coefficient
    bodyFriction = 0.2;
    bodyRestitution = 0.1;
    minImpactSpeed = 5;    // Impact
    softImpactSpeed = 5;   // Play Soft Impact Sound
    hardImpactSpeed = 15;  // Play Hard Impact Sound
    integration = 4;       // Physics integration: TickSec/Rate
    collisionTol = 0.1;    // Collision distance tolerance
    contactTol = 0.1;      // Contact velocity tolerance
    minRollSpeed = 10;

    maxDrag = 0.5;
    minDrag = 0.01;
}
dustHeight = 10;

dragForce = 0.05;
vertFactor = 0.05;
};

See also:
RigidShape
ShapeBase
Member Data Documentation

float RigidShapeData::bodyFriction

How much friction this object has. Lower values will cause the object to appear to be more slippery.

float RigidShapeData::bodyRestitution

The percentage of kinetic energy kept by this object in a collision.

float RigidShapeData::cameraDecay

Scalar rate at which the third person camera offset decays, per tick.

float RigidShapeData::cameraLag

Scalar amount by which the third person camera lags the object, relative to the object's linear velocity.

float RigidShapeData::cameraOffset

The vertical offset of the object's camera.

bool RigidShapeData::cameraRoll

Specifies whether the camera's rotation matrix, and the render eye transform are multiplied during camera updates.
**float RigidShapeData::collisionTol**

Collision distance tolerance.

**float RigidShapeData::contactTol**

Contact velocity tolerance.

**float RigidShapeData::dragForce**

Used to simulate the constant drag acting on the object.

**ParticleEmitterData RigidShapeData::dustEmitter**

Array of pointers to `ParticleEmitterData` datablocks which will be used to emit particles at object/terrain contact point.

**float RigidShapeData::dustHeight**

Height of dust effects.

**ParticleEmitterData RigidShapeData::dustTrailEmitter**

Particle emitter used to create a dust trail for the moving object.

**SFXTrack RigidShapeData::exitingWater**

The AudioProfile will be used to produce sounds when emerging from water.
**float RigidShapeData::exitSplashSoundVelocity**

The minimum velocity at which the exit splash sound will be played when emerging from water.

**SFXTrack RigidShapeData::hardImpactSound**

Sound to play when body impacts with at least hardImpactSpeed.

**float RigidShapeData::hardImpactSpeed**

Minimum speed at which the object must be travelling for the hard impact sound to be played.

**float RigidShapeData::hardSplashSoundVelocity**

The minimum velocity at which the hard splash sound will be played when impacting water.

**SFXTrack RigidShapeData::impactWaterEasy**

The AudioProfile will be used to produce sounds when a soft impact with water occurs.

**SFXTrack RigidShapeData::impactWaterHard**

The AudioProfile will be used to produce sounds when a hard impact with water occurs.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SFXTrack RigidShapeData::impactWaterMedium</code></td>
<td>The AudioProfile will be used to produce sounds when a medium impact with water occurs.</td>
</tr>
<tr>
<td><code>int RigidShapeData::integration</code></td>
<td>Number of physics steps to process per tick.</td>
</tr>
<tr>
<td><code>Point3F RigidShapeData::massBox</code></td>
<td>Size of inertial box.</td>
</tr>
<tr>
<td><code>Point3F RigidShapeData::massCenter</code></td>
<td>Center of mass for rigid body.</td>
</tr>
<tr>
<td><code>float RigidShapeData::maxDrag</code></td>
<td>Maximum drag available to this object.</td>
</tr>
<tr>
<td><code>float RigidShapeData::mediumSplashSoundVelocity</code></td>
<td>The minimum velocity at which the medium splash sound will be played when impacting water.</td>
</tr>
<tr>
<td><code>float RigidShapeData::minDrag</code></td>
<td>Minimum drag available to this object.</td>
</tr>
</tbody>
</table>
**float RigidShapeData::minImpactSpeed**

Minimum collision speed to classify collision as impact (triggers onImpact on server object).

**float RigidShapeData::minRollSpeed**

**SFXTrack RigidShapeData::softImpactSound**

Sound to play when body impacts with at least softImageSpeed but less than hardImpactSpeed.

**float RigidShapeData::softImpactSpeed**

Minimum speed at which this object must be travelling for the soft impact sound to be played.

**float RigidShapeData::softSplashSoundVelocity**

The minimum velocity at which the soft splash sound will be played when impacting water.

**ParticleEmitterData RigidShapeData::splashEmitter[2]**

Array of pointers to ParticleEmitterData datablocks which will generate splash effects.

**float RigidShapeData::splashFreqMod**

The simulated frequency modulation of a splash generated by this
object. Multiplied along with speed and time elapsed when determining splash emission rate.

**float RigidShapeData::splashVelEpsilon**

The threshold speed at which we consider the object's movement to have stopped when updating splash effects.

**float RigidShapeData::triggerDustHeight**

Maximum height from the ground at which the object will generate dust.

**float RigidShapeData::vertFactor**

The scalar applied to the vertical portion of the velocity drag acting on an object.

**SFXTrack RigidShapeData::waterWakeSound**

The AudioProfile will be used to produce sounds when a water wake is displayed.
River Class Reference

[Water]

A water volume defined by a 3D spline. More...

Inheritance diagram for River:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void <code>regenerate()</code></td>
<td>Intended as a helper to developers and editor scripts.</td>
</tr>
<tr>
<td>void <code>setBatchSize</code> (float meters)</td>
<td>Intended as a helper to developers and editor scripts.</td>
</tr>
<tr>
<td>void <code>setMaxDivisionSize</code> (float meters)</td>
<td>Intended as a helper to developers and editor scripts.</td>
</tr>
<tr>
<td>void <code>setMetersPerSegment</code> (float meters)</td>
<td>Intended as a helper to developers and editor scripts.</td>
</tr>
<tr>
<td>void <code>setNodeDepth</code> (int idx, float meters)</td>
<td>Intended as a helper to developers and editor scripts.</td>
</tr>
</tbody>
</table>
### Public Attributes

#### River

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>FlowMagnitude</td>
<td>Magnitude of the force vector applied to dynamic objects within the River.</td>
</tr>
<tr>
<td>float</td>
<td>LowLODDistance</td>
<td>Segments of the river at this distance in meters or greater will render as a single unsubdivided without undulation effects.</td>
</tr>
<tr>
<td>float</td>
<td>SegmentLength</td>
<td>Divide the River lengthwise into segments of this length in meters. These geometric volumes are used for spacial queries like determining containment.</td>
</tr>
<tr>
<td>float</td>
<td>SubdivideLength</td>
<td>For purposes of generating the renderable geometry River segments are further subdivided such that no quad is of greater width or length than this distance in meters.</td>
</tr>
</tbody>
</table>

#### Internal

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>Node</td>
<td>For internal use, do not modify.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EditorOpen</td>
<td>For editor use.</td>
</tr>
<tr>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>showNodes</td>
<td>For editor use.</td>
</tr>
<tr>
<td>showRiver</td>
<td>For editor use.</td>
</tr>
<tr>
<td>showSpline</td>
<td>For editor use.</td>
</tr>
<tr>
<td>showWalls</td>
<td>For editor use.</td>
</tr>
<tr>
<td>showWireframe</td>
<td>For editor use.</td>
</tr>
</tbody>
</table>
Detailed Description

A water volume defined by a 3D spline.

User may control width and depth per node and overall spline shape in three dimensions.

River supports dynamic planar reflections (fullReflect) like all WaterObject classes, but keep in mind it is not necessarily a planar surface. For best visual quality a River should be less reflective the more it twists and bends. This caution only applies to Rivers with fullReflect on.

See also:

WaterObject for inherited functionality.
<table>
<thead>
<tr>
<th>Member Function Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void River::regenerate()</code></td>
</tr>
<tr>
<td>Intended as a helper to developers and editor scripts. Force River to recreate its geometry.</td>
</tr>
<tr>
<td><code>void River::setBatchSize(float meters)</code></td>
</tr>
<tr>
<td>Intended as a helper to developers and editor scripts. BatchSize is not currently used.</td>
</tr>
<tr>
<td><code>void River::setMaxDivisionSize(float meters)</code></td>
</tr>
<tr>
<td>Intended as a helper to developers and editor scripts. See also: SubdivideLength field.</td>
</tr>
<tr>
<td><code>void River::setMetersPerSegment(float meters)</code></td>
</tr>
<tr>
<td>Intended as a helper to developers and editor scripts. See also: SegmentLength field.</td>
</tr>
<tr>
<td><code>void River::setNodeDepth(int idx, float meters)</code></td>
</tr>
</tbody>
</table>
Intended as a helper to developers and editor scripts.

Sets the depth in meters of a particular node.
Member Data Documentation

**bool River::EditorOpen [static]**

For editor use.

**float River::FlowMagnitude**

Magnitude of the force vector applied to dynamic objects within the River.

**float River::LowLODDistance**

Segments of the river at this distance in meters or greater will render as a single unsubdivided without undulation effects.

**string River::Node**

For internal use, do not modify.

**float River::SegmentLength**

Divide the River lengthwise into segments of this length in meters. These geometric volumes are used for spacial queries like determining containment.

**bool River::showNodes [static]**

For editor use.
bool River::showRiver [static]

For editor use.

bool River::showSpline [static]

For editor use.

bool River::showWalls [static]

For editor use.

bool River::showWireframe [static]

For editor use.

float River::SubdivideLength

For purposes of generating the renderable geometry River segments are further subdivided such that no quad is of greater width or length than this distance in meters.

Copyright © GarageGames, LLC. All Rights Reserved.
SaveFileDialog Class Reference

[File I/O]

Derived from FileDialog, this class is responsible for opening a file browser with the intention of saving a file. More...

Inheritance diagram for SaveFileDialog:

List of all members.
### Public Attributes

<table>
<thead>
<tr>
<th>bool</th>
<th>OverwritePrompt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>True/False whether the dialog should prompt before accepting an existing file name.</td>
</tr>
</tbody>
</table>
Detailed Description

Derived from FileDialog, this class is responsible for opening a file browser with the intention of saving a file.

The core usage of this dialog is to locate a file in the OS and return the path and name. This does not handle the actual file writing or data manipulation. That functionality is left up to the FileObject class.

Example:

```c++
// Create a dialog dedicated to opening files
%saveFileDlg = new SaveFileDialog()
{
    // Only allow for saving of COLLADA files
    Filters = "COLLADA Files (*.dae)|*.dae|"

    // Default save path to where the WorldEditor was last saved
    DefaultPath = $pref::WorldEditor::LastPath;

    // No default file specified
    DefaultFile = "";

    // Do not allow the user to change to a new directory
    ChangePath = false;

    // Prompt the user if they are going to overwrite an existing file
    OverwritePrompt = true;
};

// Launch the save file dialog
%saveFileDlg.Execute();

if ( %result )
{
```
%selectedFile = %openFileDialog.file;
}
else
{
    %selectedFile = "";
}

// Cleanup
%saveFileDialog.delete();

Note:

FileDialog and its related classes are only available in a Tools build of Torque.

See also:

FileDialog
OpenFileDialog
FileObject
Member Data Documentation

**bool SaveFileDialog::OverwritePrompt**

True/False whether the dialog should prompt before accepting an existing file name.

Copyright © GarageGames, LLC. All Rights Reserved.
ScatterSky Class Reference
[Atmosphere]

Represents both the sun and sky for scenes with a dynamic time of day. More...

Inheritance diagram for ScatterSky:

List of all members.
Public Member Functions

```c
void applyChanges ()
    Apply a full network update of all fields to all clients.
```
Public Attributes

**ScatterSky**

*Only azimuth and elevation are networked fields. To trigger a full update of all other fields use the applyChanges ConsoleMethod.*

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorF</td>
<td>ambientScale</td>
<td>Modulates the ambient color of sunlight.</td>
</tr>
<tr>
<td>float</td>
<td>exposure</td>
<td>Controls the contrast of the sky and sun during daytime.</td>
</tr>
<tr>
<td>ColorF</td>
<td>fogScale</td>
<td>Modulates the fog color. Note that this overrides the LevelInfo.fogColor property, so you should not use LevelInfo.fogColor if the level contains a ScatterSky object.</td>
</tr>
<tr>
<td>float</td>
<td>mieScattering</td>
<td>Affects the size and intensity of light scattering around the sun.</td>
</tr>
<tr>
<td>float</td>
<td>rayleighScattering</td>
<td>Controls how blue the atmosphere is during the day.</td>
</tr>
<tr>
<td>float</td>
<td>skyBrightness</td>
<td>Global brightness and intensity applied to the sky and objects in the level.</td>
</tr>
<tr>
<td>ColorF</td>
<td>sunScale</td>
<td>Modulates the directional color of sunlight.</td>
</tr>
</tbody>
</table>

**Advanced Lighting**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point3F</td>
<td>attenuationRatio</td>
<td>The proportions of constant, linear, and quadratic</td>
</tr>
</tbody>
</table>
attenuation to use for the falloff for point and spot lights.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filename</td>
<td>cookie</td>
</tr>
<tr>
<td></td>
<td>A custom pattern texture which is projected from the light.</td>
</tr>
<tr>
<td>float</td>
<td>fadeStartDistance</td>
</tr>
<tr>
<td></td>
<td>Start fading shadows out at this distance. 0 = auto calculate this distance.</td>
</tr>
<tr>
<td>bool</td>
<td>lastSplitTerrainOnly</td>
</tr>
<tr>
<td></td>
<td>This toggles only terrain being rendered to the last split of a PSSM shadow map.</td>
</tr>
<tr>
<td>float</td>
<td>logWeight</td>
</tr>
<tr>
<td></td>
<td>The logarithmic PSSM split distance factor.</td>
</tr>
<tr>
<td>int</td>
<td>numSplits</td>
</tr>
<tr>
<td></td>
<td>The logarithmic PSSM split distance factor.</td>
</tr>
<tr>
<td>Point4F</td>
<td>overDarkFactor</td>
</tr>
<tr>
<td></td>
<td>The ESM shadow darkening factor.</td>
</tr>
<tr>
<td>float</td>
<td>shadowDistance</td>
</tr>
<tr>
<td></td>
<td>The distance from the camera to extend the PSSM shadow.</td>
</tr>
<tr>
<td>float</td>
<td>shadowSoftness</td>
</tr>
<tr>
<td>ShadowType</td>
<td>shadowType</td>
</tr>
<tr>
<td></td>
<td>The type of shadow to use on this light.</td>
</tr>
<tr>
<td>int</td>
<td>texSize</td>
</tr>
<tr>
<td></td>
<td>The texture size of the shadow map.</td>
</tr>
</tbody>
</table>

**Orbit**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>azimuth</td>
</tr>
<tr>
<td></td>
<td>The horizontal angle of the sun measured clockwise from the positive Y world axis. This field is networked.</td>
</tr>
</tbody>
</table>
**float elevation**  
The elevation angle of the sun above or below the horizon. This field is networked.

**float moonAzimuth**  
The horizontal angle of the moon measured clockwise from the positive Y world axis. This is not animated by time or networked.

**float moonElevation**  
The elevation angle of the moon above or below the horizon. This is not animated by time or networked.

**Lighting**

**float brightness**  
The brightness of the ScatterSky's light object.

**bool castShadows**  
Enables/disables shadows cast by objects due to ScatterSky light.

**Misc**

**float flareScale**  
Changes the size and intensity of the flare.

**LightFlareData flareType**  
Datablock for the flare produced by the ScatterSky.

**Advanced Lighting Lightmap**

**bool includeLightmappedGeometryInShadow**  
This light should render lightmapped geometry during its shadow-map update (ignored if 'representedInLightmap' is false).
<table>
<thead>
<tr>
<th>Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>representedInLightmap</code></td>
<td>This light is represented in lightmaps (static light, default: false).</td>
</tr>
<tr>
<td>ColorF</td>
<td><code>shadowDarkenColor</code></td>
<td>The color that should be used to multiply-blend dynamic shadows onto lightmapped geometry (ignored if 'representedInLightmap' is false).</td>
</tr>
<tr>
<td>Night</td>
<td>bool</td>
<td><code>moonEnabled</code></td>
</tr>
<tr>
<td>ColorF</td>
<td><code>moonLightColor</code></td>
<td>Color of light cast by the directional light during night.</td>
</tr>
<tr>
<td>string</td>
<td><code>moonMat</code></td>
<td>Material for the moon sprite.</td>
</tr>
<tr>
<td>float</td>
<td><code>moonScale</code></td>
<td>Controls size the moon sprite renders, specified as a fractional amount of the screen height.</td>
</tr>
<tr>
<td>ColorF</td>
<td><code>nightColor</code></td>
<td>The ambient color during night. Also used for the sky color if <code>useNightCubemap</code> is false.</td>
</tr>
<tr>
<td>string</td>
<td><code>nightCubemap</code></td>
<td>Cubemap visible during night.</td>
</tr>
<tr>
<td>ColorF</td>
<td><code>nightFogColor</code></td>
<td>The fog color during night.</td>
</tr>
<tr>
<td>bool</td>
<td><code>useNightCubemap</code></td>
<td>Transition to the <code>nightCubemap</code> during night. If false we use <code>nightColor</code>.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>static bool</th>
<th>isRenderable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static bool</th>
<th>isSelectable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Represents both the sun and sky for scenes with a dynamic time of day.

ScatterSky renders as a dome shaped mesh which is camera relative and always overhead. It is intended to be part of the background of your scene and renders before all other objects types.

ScatterSky is designed for outdoor scenes which need to transition fluidly between radically different times of day. It will respond to time changes originating from a \texttt{TimeOfDay} object or the elevation field can be directly adjusted.

During day, ScatterSky uses atmospheric sunlight scattering approximations to generate a sky gradient and sun corona. It also calculates the fog color, ambient color, and sun color, which are used for scene lighting. This is user controlled by fields within the \texttt{ScatterSky} group.

During night, ScatterSky supports can transition to a night sky cubemap and moon sprite. The user can control this and night time colors used for scene lighting with fields within the Night group.

A scene with a \texttt{ScatterSky} should not have any other sky or sun objects as it already fulfills both roles.

ScatterSky is intended to be used with \texttt{CloudLayer} and \texttt{TimeOfDay} as part of a scene with dynamic lighting. Having a ScatterSky without a changing time of day would unnecessarily give up artistic control compared and fillrate compared to a \texttt{SkyBox + Sun} setup.
Member Function Documentation

`void ScatterSky::applyChanges()`

Apply a full network update of all fields to all clients.
Member Data Documentation

**ColorF ScatterSky::ambientScale**
Modulates the ambient color of sunlight.

**Point3F ScatterSky::attenuationRatio**
The proportions of constant, linear, and quadratic attenuation to use for the falloff for point and spot lights.

**float ScatterSky::azimuth**
The horizontal angle of the sun measured clockwise from the positive Y world axis. This field is networked.

**float ScatterSky::brightness**
The brightness of the ScatterSky's light object.

**bool ScatterSky::castShadows**
Enables/disables shadows cast by objects due to ScatterSky light.

**filename ScatterSky::cookie**
A custom pattern texture which is projected from the light.

**float ScatterSky::elevation**
The elevation angle of the sun above or below the horizon. This field is networked.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float ScatterSky::exposure</td>
<td>Controls the contrast of the sky and sun during daytime.</td>
</tr>
<tr>
<td>float ScatterSky::fadeStartDistance</td>
<td>Start fading shadows out at this distance. 0 = auto calculate this distance.</td>
</tr>
<tr>
<td>float ScatterSky::flareScale</td>
<td>Changes the size and intensity of the flare.</td>
</tr>
<tr>
<td>LightFlareData ScatterSky::flareType</td>
<td>Datablock for the flare produced by the ScatterSky.</td>
</tr>
<tr>
<td>ColorF ScatterSky::fogScale</td>
<td>Modulates the fog color. Note that this overrides the LevellInfo.fogColor property, so you should not use LevellInfo.fogColor if the level contains a ScatterSky object.</td>
</tr>
<tr>
<td>bool ScatterSky::includeLightmappedGeometryInShadow</td>
<td>This light should render lightmapped geometry during its shadow-map update (ignored if 'representedInLightmap' is false).</td>
</tr>
<tr>
<td>Variable</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td></td>
</tr>
<tr>
<td><strong>bool ScatterSky::lastSplitTerrainOnly</strong></td>
<td></td>
</tr>
<tr>
<td>This toggles only terrain being rendered to the last split of a PSSM shadow map.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>float ScatterSky::logWeight</strong></td>
</tr>
<tr>
<td>The logarithmic PSSM split distance factor.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>float ScatterSky::mieScattering</strong></td>
</tr>
<tr>
<td>Affects the size and intensity of light scattering around the sun.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>float ScatterSky::moonAzimuth</strong></td>
</tr>
<tr>
<td>The horizontal angle of the moon measured clockwise from the positive Y world axis. This is not animated by time or networked.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>float ScatterSky::moonElevation</strong></td>
</tr>
<tr>
<td>The elevation angle of the moon above or below the horizon. This is not animated by time or networked.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>bool ScatterSky::moonEnabled</strong></td>
</tr>
<tr>
<td>Enable or disable rendering of the moon sprite during night.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ColorF ScatterSky::moonLightColor</strong></td>
</tr>
</tbody>
</table>
Color of light cast by the directional light during night.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string ScatterSky::moonMat</td>
<td>Material for the moon sprite.</td>
</tr>
<tr>
<td>float ScatterSky::moonScale</td>
<td>Controls size the moon sprite renders, specified as a fractional amount of the screen height.</td>
</tr>
<tr>
<td>ColorF ScatterSky::nightColor</td>
<td>The ambient color during night. Also used for the sky color if useNightCubemap is false.</td>
</tr>
<tr>
<td>string ScatterSky::nightCubemap</td>
<td>Cubemap visible during night.</td>
</tr>
<tr>
<td>ColorF ScatterSky::nightFogColor</td>
<td>The fog color during night.</td>
</tr>
<tr>
<td>int ScatterSky::numSplits</td>
<td>The logarithmic PSSM split distance factor.</td>
</tr>
<tr>
<td>Point4F ScatterSky::overDarkFactor</td>
<td></td>
</tr>
</tbody>
</table>
The ESM shadow darkening factor.

<table>
<thead>
<tr>
<th>float ScatterSky::rayleighScattering</th>
</tr>
</thead>
</table>

Controls how blue the atmosphere is during the day.

<table>
<thead>
<tr>
<th>bool ScatterSky::representedInLightmap</th>
</tr>
</thead>
</table>

This light is represented in lightmaps (static light, default: false).

<table>
<thead>
<tr>
<th>ColorF ScatterSky::shadowDarkenColor</th>
</tr>
</thead>
</table>

The color that should be used to multiply-blend dynamic shadows onto lightmapped geometry (ignored if 'representedInLightmap' is false).

<table>
<thead>
<tr>
<th>float ScatterSky::shadowDistance</th>
</tr>
</thead>
</table>

The distance from the camera to extend the PSSM shadow.

<table>
<thead>
<tr>
<th>float ScatterSky::shadowSoftness</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>ShadowType ScatterSky::shadowType</th>
</tr>
</thead>
</table>

The type of shadow to use on this light.

<table>
<thead>
<tr>
<th>float ScatterSky::skyBrightness</th>
</tr>
</thead>
</table>
Global brightness and intensity applied to the sky and objects in the level.

<table>
<thead>
<tr>
<th>ColorF ScatterSky::sunScale</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modulates the directional color of sunlight.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int ScatterSky::texSize</th>
</tr>
</thead>
<tbody>
<tr>
<td>The texture size of the shadow map.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool ScatterSky::useNightCubemap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transition to the nightCubemap during night. If false we use nightColor.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
SceneObject Class Reference
[Game Objects]

A networkable object that exists in the 3D world. More...

Inheritance diagram for SceneObject:
List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point3F</td>
<td><code>getEulerRotation()</code></td>
<td>Get Euler rotation of this object.</td>
</tr>
<tr>
<td>VectorF</td>
<td><code>getForwardVector()</code></td>
<td>Get the direction this object is facing.</td>
</tr>
<tr>
<td>TransformF</td>
<td><code>getInverseTransform()</code></td>
<td>Get the object's inverse transform.</td>
</tr>
<tr>
<td>int</td>
<td><code>getMountedObject(int slot)</code></td>
<td>Get the object mounted at a particular slot.</td>
</tr>
<tr>
<td>int</td>
<td><code>getMountedObjectCount()</code></td>
<td>Get the number of objects mounted to us.</td>
</tr>
<tr>
<td>int</td>
<td><code>getMountedObjectNode(int slot)</code></td>
<td>Get the mount node index of the object mounted at our given slot.</td>
</tr>
<tr>
<td>int</td>
<td><code>getMountNodeObject(int node)</code></td>
<td>Get the object mounted at our given node index.</td>
</tr>
<tr>
<td>Box3F</td>
<td><code>getObjectBox()</code></td>
<td>Get the object's bounding box (relative to the object's origin).</td>
</tr>
<tr>
<td>int</td>
<td><code>getObjectMount()</code></td>
<td>Get the object we are mounted to.</td>
</tr>
<tr>
<td>Point3F</td>
<td><code>GetPosition()</code></td>
<td>Get the object's world position.</td>
</tr>
<tr>
<td>VectorF</td>
<td><code>getRightVector()</code></td>
<td>Get the right vector of the object.</td>
</tr>
<tr>
<td>Point3F</td>
<td><code>getScale()</code></td>
<td>Get the object's scale.</td>
</tr>
<tr>
<td>TransformF</td>
<td><code>getTransform()</code></td>
<td>Get the object's transform.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><code>int getType()</code></td>
<td>Return the type mask for this object.</td>
<td></td>
</tr>
<tr>
<td><code>VectorF getUpVector()</code></td>
<td>Get the up vector of the object.</td>
<td></td>
</tr>
<tr>
<td><code>Box3F getWorldBox()</code></td>
<td>Get the object's world bounding box.</td>
<td></td>
</tr>
<tr>
<td><code>Point3F getWorldBoxCenter()</code></td>
<td>Get the center of the object's world bounding box.</td>
<td></td>
</tr>
<tr>
<td><code>bool isGlobalBounds()</code></td>
<td>Check if this object has a global bounds set.</td>
<td></td>
</tr>
<tr>
<td><code>bool isMounted()</code></td>
<td>Check if we are mounted to another object.</td>
<td></td>
</tr>
<tr>
<td><code>bool mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</code></td>
<td>Mount objB to this object at the desired slot with optional transform.</td>
<td></td>
</tr>
<tr>
<td><code>void setScale(Point3F scale)</code></td>
<td>Set the object's scale.</td>
<td></td>
</tr>
<tr>
<td><code>void setTransform(TransformF txfm)</code></td>
<td>Set the object's transform (orientation and position).</td>
<td></td>
</tr>
<tr>
<td><code>void unmount()</code></td>
<td>Unmount us from the currently mounted object if any.</td>
<td></td>
</tr>
<tr>
<td><code>bool unmountObject(SceneObject target)</code></td>
<td>Unmount an object from ourselves.</td>
<td></td>
</tr>
</tbody>
</table>
Public Attributes

Editing

<table>
<thead>
<tr>
<th>bool</th>
<th>isRenderEnabled</th>
<th>Controls client-side rendering of the object.</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>isSelectionEnabled</td>
<td>Determines if the object may be selected from within the Tools.</td>
</tr>
</tbody>
</table>

Mounting

<table>
<thead>
<tr>
<th>int</th>
<th>mountNode</th>
<th>Node we are mounted to.</th>
</tr>
</thead>
<tbody>
<tr>
<td>pid</td>
<td>mountPID</td>
<td>PersistentID of object we are mounted to.</td>
</tr>
<tr>
<td></td>
<td>mountPos</td>
<td>Position we are mounted at (object space of our mount object).</td>
</tr>
<tr>
<td></td>
<td>mountRot</td>
<td>Rotation we are mounted at (object space of our mount object).</td>
</tr>
</tbody>
</table>

Transform

<table>
<thead>
<tr>
<th>MatrixPosition</th>
<th>position</th>
<th>Object world position.</th>
</tr>
</thead>
<tbody>
<tr>
<td>MatrixRotation</td>
<td>rotation</td>
<td>Object world orientation.</td>
</tr>
<tr>
<td>Point3F</td>
<td>scale</td>
<td>Object world scale.</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>static bool</th>
<th>isRenderable</th>
<th>Disables rendering of all instances of this type.</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

A networkable object that exists in the 3D world.

The SceneObject class provides the foundation for 3D objects in the Engine. It exposes the functionality for:

- Position, rotation and scale within the world.
- Working with a scene graph (in the Zone and Portal sections), allowing efficient and robust rendering of the game scene.
- Various helper functions, including functions to get bounding information and momentum/velocity.
- Mounting one SceneObject to another.
- An interface for collision detection, as well as ray casting.
- Lighting. SceneObjects can register lights both at lightmap generation time, and dynamic lights at runtime (for special effects, such as from flame or a projectile, or from an explosion).

You do not typically work with SceneObjects themselves. The SceneObject provides a reference within the game world (the scene), but does not render to the client on its own. The same is true of collision detection beyond that of the bounding box. Instead you use one of the many classes that derive from SceneObject, such as TSStatic.
Difference Between setHidden() and isRenderEnabled

When it comes time to decide if a SceneObject should render or not, there are two methods that can stop the SceneObject from rendering at all. You need to be aware of the differences between these two methods as they impact how the SceneObject is networked from the server to the client.

The first method of manually controlling if a SceneObject is rendered is through its SceneObject::isRenderEnabled property. When set to false the SceneObject is considered invisible but still present within the scene. This means it still takes part in collisions and continues to be networked.

The second method is using the setHidden() method. This will actually remove a SceneObject from the scene and it will no longer be networked from the server to the client. Any client-side ghost of the object will be deleted as the server no longer considers the object to be in scope.
Member Function Documentation

**Point3F SceneObject::getEulerRotation()**

Get Euler rotation of this object.

**Returns:**
the orientation of the object in the form of rotations around the X, Y and Z axes in degrees.

**VectorF SceneObject::getForwardVector()**

Get the direction this object is facing.

**Returns:**
a vector indicating the direction this object is facing.

**Note:**
This is the object's y axis.

**TransformF SceneObject::getInverseTransform()**

Get the object's inverse transform.

**Returns:**
the inverse transform of the object

**int SceneObject::getMountedObject(int slot)**

Get the object mounted at a particular slot.

**Parameters:**
slot mount slot index to query

Returns:
ID of the object mounted in the slot, or 0 if no object.

```cpp
int SceneObject::getMountedObjectCount()
```

Get the number of objects mounted to us.

Returns:
the number of mounted objects.

```cpp
int SceneObject::getMountedObjectNode(int slot)
```

Get the mount node index of the object mounted at our given slot.

Parameters:
slot mount slot index to query

Returns:
index of the mount node used by the object mounted in this slot.

```cpp
int SceneObject::getMountNodeObject(int node)
```

Get the object mounted at our given node index.

Parameters:
node mount node index to query

Returns:
ID of the first object mounted at the node, or 0 if none found.
Box3F SceneObject::getObjectBox()

Get the object's bounding box (relative to the object's origin).

**Returns:**

six fields, two Point3Fs, containing the min and max points of the object box.

int SceneObject::getObjectMount()

Get the object we are mounted to.

**Returns:**

the SimObjectID of the object we're mounted to, or 0 if not mounted.

Point3F SceneObject::getPosition()

Get the object's world position.

**Returns:**

the current world position of the object

Reimplemented in Camera.

VectorF SceneObject::getRightVector()

Get the right vector of the object.

**Returns:**

a vector indicating the right direction of this object.

**Note:**
This is the object's x axis.

**Point3F SceneObject::getScale( )**

Get the object's scale.

**Returns:**
object scale as a Point3F

**TransformF SceneObject::getTransform( )**

Get the object's transform.

**Returns:**
the current transform of the object

**int SceneObject::getType( )**

Return the type mask for this object.

**Returns:**
The numeric type mask for the object.

**VectorF SceneObject::getUpVector( )**

Get the up vector of the object.

**Returns:**
a vector indicating the up direction of this object.

**Note:**
This is the object's z axis.
Box3F SceneObject::getWorldBox()

Get the object's world bounding box.

**Returns:**

six fields, two Point3Fs, containing the min and max points of the worldbox.

Point3F SceneObject::getWorldBoxCenter()

Get the center of the object's world bounding box.

**Returns:**

the center of the world bounding box for this object.

bool SceneObject::isGlobalBounds()

Check if this object has a global bounds set.

If global bounds are set to be true, then the object is assumed to have an infinitely large bounding box for collision and rendering purposes.

**Returns:**

true if the object has a global bounds.

bool SceneObject::isMounted()

Check if we are mounted to another object.

**Returns:**

true if mounted to another object, false if not mounted.
bool SceneObject::mountObject(SceneObject objB, int slot, TransformF txfm = MatrixF::Identity)

Mount objB to this object at the desired slot with optional transform.

Parameters:
- objB Object to mount onto us
- slot Mount slot ID
- txfm (optional) mount offset transform

Returns:
true if successful, false if failed (objB is not valid)

void SceneObject::setScale(Point3F scale)

Set the object's scale.

Parameters:
- scale object scale to set

void SceneObject::setTransform(TransformF txfm)

Set the object's transform (orientation and position).

Parameters:
- txfm object transform to set

void SceneObject::unmount()
Unmount us from the currently mounted object if any.

bool SceneObject::unmountObject(SceneObject target)

Unmount an object from ourselves.

**Parameters:**

target object to unmount

**Returns:**

true if successful, false if failed
### Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Member</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>SceneObject::isRenderEnabled</td>
</tr>
</tbody>
</table>

Controls client-side rendering of the object.

**See also:**
- isRenderable()

| bool          | SceneObject::isSelectionEnabled             |

Determines if the object may be selected from within the Tools.

**See also:**
- isSelectable()

| int           | SceneObject::mountNode                     |

Node we are mounted to.

| pid           | SceneObject::mountPID                      |

PersistentID of object we are mounted to.

Unlike the SimObjectID that is determined at run time, the PersistentID of an object is saved with the level/mission and may be used to form a link between objects.

| MatrixPosition | SceneObject::mountPos                     |

Position we are mounted at (object space of our mount object).
<table>
<thead>
<tr>
<th><strong>MatrixRotation SceneObject::mountRot</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotation we are mounted at (object space of our mount object).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>MatrixPosition SceneObject::position</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Object world position.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>MatrixRotation SceneObject::rotation</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Object world orientation.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Point3F SceneObject::scale</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Object world scale.</td>
</tr>
</tbody>
</table>
ScriptGroup Class Reference
[Console, Scripting]

Essentially a SimGroup, but with onAdd and onRemove script callbacks. More...

Inheritance diagram for ScriptGroup:

```
SimObject
  ▼
  ▼
SimSet
  ▼
  ▼
SimGroup
  ▼
  ▼
ScriptGroup
```

List of all members.
## Public Member Functions

### Callbacks

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void onAdd (SimObjectId ID)</code></td>
<td>Called when this <code>ScriptGroup</code> is added to the system.</td>
</tr>
<tr>
<td><code>void onRemove (SimObjectId ID)</code></td>
<td>Called when this <code>ScriptObject</code> is removed from the system.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Essentially a SimGroup, but with onAdd and onRemove script callbacks.

**Example:**

```plaintext
// First container, SimGroup containing a:
new SimGroup(Scenes)
{
    // Subcontainer, ScriptGroup containing
    // related to a cut scene and a starting
    new ScriptGroup(WelcomeScene)
    {
        class = "Scene";
        pathName = "Pathx";
        description = "A small orc village set in the Hardesty mountains. This town and its surroundings will be used to illustrate some of the Torque Game Engine's features."
        pathTime = "0";
        title = "Welcome to Orc Town";

        new WayPoint(start)
        {
            position = "163.873 -103.82 208.354";
            rotation = "0.136165 -0.0544916 0.989186 44.0527";
            scale = "1 1 1";
            dataBlock = "WayPointMarker";
            team = "0";
        }
    }
};
```

**See also:**

SimGroup
**Member Function Documentation**

```cpp
void ScriptGroup::onAdd (SimObjectId ID )
```

Called when this `ScriptGroup` is added to the system.

**Parameters:**

- `ID` Unique object ID assigned when created (this in script).

```cpp
void ScriptGroup::onRemove (SimObjectId ID )
```

Called when this `ScriptObject` is removed from the system.

**Parameters:**

- `ID` Unique object ID assigned when created (this in script).

Copyright © GarageGames, LLC. All Rights Reserved.
ScriptMsgListener Class Reference
[Messaging]

Script accessible version of Dispatcher::IMessageListener. Often used in conjunction with EventManager. More...

Inheritance diagram for ScriptMsgListener:

```
SimObject

ScriptMsgListener

MessageForwarder
```

List of all members.
**Public Member Functions**

**Callbacks**

<table>
<thead>
<tr>
<th>Function Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>onAdd ()</td>
<td>Script callback when a listener is first created and registered.</td>
</tr>
<tr>
<td>void</td>
<td>onAddToQueue (string queue)</td>
<td>Callback for when the listener is added to a queue.</td>
</tr>
<tr>
<td>bool</td>
<td>onMessageObjectReceived (string queue, Message msg)</td>
<td>Called when a message object (not just the message data) is passed to a listener.</td>
</tr>
<tr>
<td>bool</td>
<td>onMessageReceived (string queue, string event, string data)</td>
<td>Called when the listener has received a message.</td>
</tr>
<tr>
<td>void</td>
<td>onRemove ()</td>
<td>Script callback when a listener is deleted.</td>
</tr>
<tr>
<td>void</td>
<td>onRemoveFromQueue (string queue)</td>
<td>Callback for when the listener is removed from a queue.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Script accessible version of Dispatcher::IMessageListener. Often used in conjunction with EventManager.

The main use of ScriptMsgListener is to allow script to listen formessages. You can subclass ScriptMsgListener in script to receivethe Dispatcher::IMessageListener callbacks.

Alternatively, you can derive from it in C++ instead of SimObject to get an object that implements Dispatcher::IMessageListener with script callbacks. If you need to derive from something other then SimObject, then you will need to implement the Dispatcher::IMessageListener interface yourself.

**Example:**

```cpp
// Create the EventManager.
$MyEventManager = new EventManager() {
    queue =
    // Create an event.
    $MyEventManager.registerEvent( "SomeCoolEvent"
    // Create a listener and subscribe.
    $MyListener = new ScriptMsgListener() {
        class $MyEventManager.subscribe( $MyListener, "SomeCoolEvent" function MyListener::onSomeCoolEvent( %this
        { echo( "onSomeCoolEvent Triggered" );
        }
    // Trigger the event.
    $MyEventManager.postEvent( "SomeCoolEvent"
```
Member Function Documentation

void ScriptMsgListener::onAdd()

Script callback when a listener is first created and registered.

Example:

```cpp
function ScriptMsgListener::onAdd(%this)
{
    // Perform on add code here
}
```

void ScriptMsgListener::onAddToQueue(string queue)

Callback for when the listener is added to a queue.

The default implementation of `onAddToQueue()` and `onRemoveFromQueue()` provide tracking of the queues this listener is added to through the mQueues member. Overrides of `onAddToQueue()` or `onRemoveFromQueue()` should ensure they call the parent implementation in any overrides.

**Parameters:**

- `queue` The name of the queue that the listener added to

**See also:**

- `onRemoveFromQueue()`

bool ScriptMsgListener::onMessageObjectReceived(string queue, Message msg)

Called when a message object (not just the message data) is
passed to a listener.

**Parameters:**

- `queue` The name of the queue the message was dispatched to
- `msg` The message object

**Returns:**

false to prevent other listeners receiving this message, true otherwise

**See also:**

Message
onMessageReceived

```cpp
bool ScriptMsgListener::onMessageReceived(string queue,
                                         string event,
                                         string data)
```

Called when the listener has received a message.

**Parameters:**

- `queue` The name of the queue the message was dispatched to
- `event` The name of the event (function) that was triggered
- `data` The data (parameters) for the message

**Returns:**

false to prevent other listeners receiving this message, true otherwise

```cpp
void ScriptMsgListener::onRemove()
```
Script callback when a listener is deleted.

Example:

```cpp
function ScriptMsgListener::onRemove(%this)
{
    // Perform on remove code here
}
```

```cpp
void ScriptMsgListener::onRemoveFromQueue(string queue)
```

Callback for when the listener is removed from a queue.

The default implementation of `onAddToQueue()` and `onRemoveFromQueue()` provide tracking of the queues this listener is added to through the mQueues member. Overrides of `onAddToQueue()` or `onRemoveFromQueue()` should ensure they call the parent implementation in any overrides.

**Parameters:**

- `queue` The name of the queue that the listener was removed from.

**See also:**

- `onAddToQueue()`

Copyright © GarageGames, LLC. All Rights Reserved.
ScriptObject Class Reference
[Console, Scripting]

A script-level OOP object which allows binding of a class, superClass and arguments along with declaration of methods.

More...

Inheritance diagram for ScriptObject:

List of all members.
Public Member Functions

Callbacks

<table>
<thead>
<tr>
<th>void</th>
<th>onAdd (SimObjectId ID)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Called when this ScriptObject is added to the system.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>onRemove (SimObjectId ID)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Called when this ScriptObject is removed from the system.</td>
</tr>
</tbody>
</table>
**Detailed Description**

A script-level OOP object which allows binding of a class, superClass and arguments along with declaration of methods.

ScriptObjects are extraordinarily powerful objects that allow defining of any type of data required. They can optionally have a class and a superclass defined for added control of multiple ScriptObjects through a simple class definition.

**Example:**

```javascript
new ScriptObject(Game) {
    class = "DeathMatchGame";
    superClass = GameCore;
    genre = "Action FPS"; // Note the new,
};
```

**See also:**

SimObject
Member Function Documentation

void ScriptObject::onAdd (SimObjectId ID )

Called when this ScriptObject is added to the system.

Parameters:

\(ID\) Unique object ID assigned when created (this in script).

void ScriptObject::onRemove (SimObjectId ID )

Called when this ScriptObject is removed from the system.

Parameters:

\(ID\) Unique object ID assigned when created (this in script).

Copyright © GarageGames, LLC. All Rights Reserved.
SFXAmbience Class Reference

[Sound]

A datablock that describes an ambient sound space. More...

Inheritance diagram for SFXAmbience:

SimObject

SimDataBlock

SFXAmbience

[legend]

List of all members.
## Public Attributes

### Sound

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>dopplerFactor</td>
<td>The factor to apply to the doppler affect in this space.</td>
</tr>
<tr>
<td>SFXEnvironment</td>
<td>environment</td>
<td>Reverb environment active in the ambience zone.</td>
</tr>
<tr>
<td>float</td>
<td>rolloffFactor</td>
<td>The rolloff factor to apply to distance-based volume attenuation in this space.</td>
</tr>
<tr>
<td>SFXTrack</td>
<td>soundTrack</td>
<td>Sound track to play in the ambience zone.</td>
</tr>
<tr>
<td>SFXState</td>
<td>states [4]</td>
<td>States to activate when the ambient zone is entered.</td>
</tr>
</tbody>
</table>
Detailed Description

A datablock that describes an ambient sound space.

Each ambience datablock captures the properties of a unique ambient sound space. A sound space is comprised of:

- an ambient audio track that is played when the listener is inside the space,
- a reverb environment that is active inside the space, and
- a number of SFXStates that are activated when entering the space and deactivated when exiting it.

Each of these properties is optional.

An important characteristic of ambient audio spaces is that their unique nature is not determined by their location in space but rather by their SFXAmbience datablock. This means that the same SFXAmbience datablock assigned to multiple locations in a level represents the same unique audio space to the sound system.

This is an important distinction for the ambient sound mixer which will activate a given ambient audio space only once at any one time regardless of how many intersecting audio spaces with the same SFXAmbience datablock assigned the listener may currently be in.

All SFXAmbience instances are automatically added to the global SFXAmbienceSet.

At the moment, transitions between reverb environments are not blended and different reverb environments from multiple active SFXAmbiences will not be blended together. This will be added in a future version.

Example:

```
singleton SFXAmbience( Underwater )
{
    environment = AudioEnvUnderwater;
```
soundTrack = ScubaSoundList;
states[ 0 ] = AudioLocationUnderwater;
};

See also:

SFXEnvironment
SFXTrack
SFXState
LevelInfo::soundAmbience
Zone::soundAmbience

Datablocks and Networking
## Member Data Documentation

### float SFXAmbience::dopplerFactor

The factor to apply to the doppler affect in this space.

Defaults to 0.5.

**Doppler Effect**

### SFXEnvironment SFXAmbience::environment

Reverb environment active in the ambience zone.

**Audio Reverb**

### float SFXAmbience::rolloffFactor

The rolloff factor to apply to distance-based volume attenuation in this space.

Defaults to 1.0.

**Note:**
This applies to the logarithmic distance model only.

**Volume Attenuation**

### SFXTrack SFXAmbience::soundTrack

Sound track to play in the ambience zone.
States to activate when the ambient zone is entered.

When the ambient sound state is entered, all states associated with the state will be activated (given that they are not disabled) and deactivated when the space is exited again.
SFXController Class Reference
[Sound]

A sound source that drives multi-source playback. More...

Inheritance diagram for SFXController:

List of all members.
Public Member Functions

```plaintext
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int getCurrentSlot ()</td>
<td>Get the index of the playlist slot currently processed by the controller.</td>
</tr>
<tr>
<td>void setCurrentSlot (int index)</td>
<td>Set the index of the playlist slot to play by the controller. This can be used to seek in the playlist.</td>
</tr>
</tbody>
</table>
```
### Public Attributes

#### Debug

<table>
<thead>
<tr>
<th>bool</th>
<th>trace</th>
</tr>
</thead>
</table>
|       | If true, the controller logs its operation to the console.
Detailed Description

A sound source that drives multi-source playback.

This class acts as an interpreter for SFXPlayLists. It goes through the slots of the playlist it is attached to and performs the actions described by each of the slots in turn. As SFXControllers are created implicitly by the SFX system when instantiating a source for a playlist it is in most cases not necessary to directly deal with the class. The following example demonstrates how a controller would commonly be created.

Example:

```java
// Create a play list from two SFXProfiles
%playList = new SFXPlayList()
{
    // Use a looped description so the list
    description = AudioMusicLoop2D;

    track[ 0 ] = Profile1;
    track[ 1 ] = Profile2;
};

// Play the list. This will implicitly create
sfxPlayOnce( %playList );
```

Note:

Play lists are updated at regular intervals by the sound system. This processing determines the granularity at which playlist action timing takes place.

This class cannot be instantiated directly. Use sfxPlayOnce() or sfxCreateSource() with the playlist you want to play to create an instance of this class.

See also:
SFXPlayList
Member Function Documentation

```c++
int SFXController::getCurrentSlot()
```

Get the index of the playlist slot currently processed by the controller.

**Returns:**

The slot index currently being played.

**See also:**

`SFXPlayList`

```c++
void SFXController::setCurrentSlot(int index)
```

Set the index of the playlist slot to play by the controller. This can be used to seek in the playlist.

**Parameters:**

`index` Index of the playlist slot.
Member Data Documentation

bool SFXController::trace

If true, the controller logs its operation to the console.

This is a non-networked field that will work locally only.
SFXDescription Class Reference
[Sound]

A description for how a sound should be played. More...

Inheritance diagram for SFXDescription:

![Inheritance diagram]

List of all members.
## Public Attributes

### 3D

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>coneInsideAngle</td>
<td>Inner sound cone angle in degrees.</td>
</tr>
<tr>
<td>int</td>
<td>coneOutsideAngle</td>
<td>Outer sound cone angle in degrees.</td>
</tr>
<tr>
<td>float</td>
<td>coneOutsideVolume</td>
<td>Determines the volume scale factor applied to a source's base volume level outside of the outer cone.</td>
</tr>
<tr>
<td>bool</td>
<td>is3D</td>
<td>If true, sounds played with this description will have a position and orientation in space.</td>
</tr>
<tr>
<td>float</td>
<td>maxDistance</td>
<td>The distance at which attenuation stops.</td>
</tr>
<tr>
<td>float</td>
<td>referenceDistance</td>
<td>Distance at which volume attenuation begins.</td>
</tr>
<tr>
<td>float</td>
<td>rolloffFactor</td>
<td>Scale factor to apply to logarithmic distance attenuation curve. If -1, the global rolloff setting is used.</td>
</tr>
<tr>
<td>Point3F</td>
<td>scatterDistance</td>
<td>Bounds on random displacement of 3D sound positions.</td>
</tr>
</tbody>
</table>

### Fading

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EaseF</td>
<td>fadeInEase</td>
<td>Easing curve for fade-in transition.</td>
</tr>
<tr>
<td>float</td>
<td>fadeInTime</td>
<td>Number of seconds to gradually fade in volume from</td>
</tr>
</tbody>
</table>
zero when playback starts.

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>fadeLoops</td>
<td>Fade each cycle of a loop in and/or out; otherwise only fade-in first cycle.</td>
</tr>
<tr>
<td>EaseF</td>
<td>fadeOutEase</td>
<td>Easing curve for fade-out transition.</td>
</tr>
<tr>
<td>float</td>
<td>fadeOutTime</td>
<td>Number of seconds to gradually fade out volume down to zero when playback is stopped or paused.</td>
</tr>
</tbody>
</table>

**Playback**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>isLooping</td>
<td>If true, the sound will be played in an endless loop.</td>
</tr>
<tr>
<td>string</td>
<td>parameters</td>
<td>Names of the parameters to which sources using this description will automatically be linked.</td>
</tr>
<tr>
<td>float</td>
<td>pitch</td>
<td>Pitch shift to apply to playback.</td>
</tr>
<tr>
<td>float</td>
<td>priority</td>
<td>Priority level for virtualization of sounds (1 = base level).</td>
</tr>
<tr>
<td>SFXSource</td>
<td>sourceGroup</td>
<td>Group that sources playing with this description should be put into.</td>
</tr>
<tr>
<td>bool</td>
<td>useHardware</td>
<td>Whether the sound is allowed to be mixed in hardware.</td>
</tr>
<tr>
<td>float</td>
<td>volume</td>
<td>Base volume level for the sound.</td>
</tr>
</tbody>
</table>

**Streaming**
**bool** `isStreaming`
If true, incrementally stream sounds; otherwise sounds are loaded in full.

**int** `streamPacketSize`
Number of seconds of sample data per single streaming packet.

**int** `streamReadAhead`
Number of sample packets to read and buffer in advance.

**Reverb**

**float** `reverbAirAbsorptionFactor`
Multiplies SFXEnvironment::airAbsorptionHR.

**int** `reverbDirect`
Direct path level (at low and mid frequencies).

**int** `reverbDirectHF`
Relative direct path level at high frequencies.

**float** `reverbDopplerFactor`
Per-source doppler factor.

**int** `reverbExclusion`
Main exclusion control (attenuation at high frequencies).

**float** `reverbExclusionLFRatio`
Exclusion low-frequency level re. main control.

**int** `reverbFlags`
Bitfield combination of per-sound reverb flags.

**int** `reverbObstruction`
Main obstruction control (attenuation at high frequencies).

**float** `reverbObstructionLFRatio`
Obstruction low-frequency level re. main control.
<table>
<thead>
<tr>
<th>Data Type</th>
<th>variable name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>int</code></td>
<td><code>reverbOcclusion</code></td>
<td>Main occlusion control (attenuation at high frequencies).</td>
</tr>
<tr>
<td><code>float</code></td>
<td><code>reverbOcclusionDirectRatio</code></td>
<td>Relative occlusion control for direct path.</td>
</tr>
<tr>
<td><code>float</code></td>
<td><code>reverbOcclusionLFRatio</code></td>
<td>Occlusion low-frequency level re. main control.</td>
</tr>
<tr>
<td><code>float</code></td>
<td><code>reverbOcclusionRoomRatio</code></td>
<td>Relative occlusion control for room effect.</td>
</tr>
<tr>
<td><code>int</code></td>
<td><code>reverbOutsideVolumeHF</code></td>
<td>Outside sound cone level at high frequencies.</td>
</tr>
<tr>
<td><code>float</code></td>
<td><code>reverbReverbRolloffFactor</code></td>
<td>Per-source logarithmic falloff factor.</td>
</tr>
<tr>
<td><code>int</code></td>
<td><code>reverbRoom</code></td>
<td>Room effect level (at low and mid frequencies).</td>
</tr>
<tr>
<td><code>int</code></td>
<td><code>reverbRoomHF</code></td>
<td>Relative room effect level at high frequencies.</td>
</tr>
<tr>
<td><code>float</code></td>
<td><code>reverbRoomRolloffFactor</code></td>
<td>Room effect falloff factor.</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><code>useCustomReverb</code></td>
<td>If true, use the reverb properties defined here on sounds.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>REVERB_DIRECTHFAUTO</td>
<td>static const int</td>
<td>Automatic setting of SFXDescription::reverbDirect due to distance to listener.</td>
</tr>
<tr>
<td>REVERB_INSTANCE0</td>
<td>static const int</td>
<td>EAX4/SFX/GameCube/Wii: Specify channel to target reverb instance 0. Default target.</td>
</tr>
<tr>
<td>REVERB_INSTANCE1</td>
<td>static const int</td>
<td>EAX4/SFX/GameCube/Wii: Specify channel to target reverb instance 1.</td>
</tr>
<tr>
<td>REVERB_INSTANCE2</td>
<td>static const int</td>
<td>EAX4/SFX/GameCube/Wii: Specify channel to target reverb instance 2.</td>
</tr>
<tr>
<td>REVERB_INSTANCE3</td>
<td>static const int</td>
<td>EAX4/SFX/GameCube/Wii: Specify channel to target reverb instance 3.</td>
</tr>
<tr>
<td>REVERB_ROOMAUTO</td>
<td>static const int</td>
<td>Automatic setting of SFXDescription::reverbRoom due to distance to listener.</td>
</tr>
<tr>
<td>REVERB_ROOMHFAUTO</td>
<td>static const int</td>
<td>Automatic setting of SFXDescription::reverbRoomHF due to distance to listener.</td>
</tr>
</tbody>
</table>
Detailed Description

A description for how a sound should be played.

SFXDescriptions are used by the sound system to collect all parameters needed to set up a given sound for playback. This includes information like its volume level, its pitch shift, etc. as well as more complex information like its fade behavior, 3D properties, and per-sound reverb properties.

Any sound playback will require a valid SFXDescription.

As datablocks, SFXDescriptions can be set up as either networked datablocks or non-networked datablocks, though it generally makes sense to keep all descriptions non-networked since they will be used exclusively by clients.

Example:

```plaintext
// A description for a 3D sound with a reasonable default range setting.
// The description is set up to assign sounds to the AudioChannelEffects source group
// (defined in the core scripts). An alternative means to achieve this is to use the AudioEffects description as a copy source.

singleton SFXDescription( Audio3DSound )
{
    sourceGroup = AudioChannelEffects;
    is3D = true;
    referenceDistance = 20.0;
    maxDistance = 100.0;
};
```
### Member Data Documentation

**int SFXDescription::coneInsideAngle**

Inner sound cone angle in degrees.

This value determines the angle of the inner volume cone that protrudes out in the direction of a sound. Within this cone, the sound source retains full volume that is unaffected by sound cone settings (though still affected by distance attenuation.)

Valid values are from 0 to 360. Must be less than coneOutsideAngle. Default is 360. Only for 3D sounds.

---

**int SFXDescription::coneOutsideAngle**

Outer sound cone angle in degrees.

This value determines the angle of the outer volume cone that protrudes out in the direction of a sound and surrounds the inner volume cone. Within this cone, volume will linearly interpolate from the outer cone hull inwards to the inner coner hull starting with the base volume scaled by coneOutsideVolume and ramping up/down to the full base volume.

Valid values are from 0 to 360. Must be >= coneInsideAngle. Default is 360. Only for 3D sounds.

---

**float SFXDescription::coneOutsideVolume**

Determines the volume scale factor applied the a source's base
volume level outside of the outer cone.

In the outer cone, starting from outside the inner cone, the scale factor smoothly interpolates from 1.0 (within the inner cone) to this value. At the moment, the allowed range is 0.0 (silence) to 1.0 (no attenuation) as amplification is only supported on XAudio2 but not on the other devices.

Only for 3D sound.

**Sound Cones**

<table>
<thead>
<tr>
<th><strong>EaseF SFXDescription::fadeInEase</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Easing curve for fade-in transition.</td>
</tr>
<tr>
<td>Volume fade-ins will interpolate volume along this curve.</td>
</tr>
</tbody>
</table>

**Volume Fades**

<table>
<thead>
<tr>
<th><strong>float SFXDescription::fadeInTime</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of seconds to gradually fade in volume from zero when playback starts.</td>
</tr>
<tr>
<td>Must be &gt;= 0.</td>
</tr>
</tbody>
</table>

**Volume Fades**

<table>
<thead>
<tr>
<th><strong>bool SFXDescription::fadeLoops</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Fade each cycle of a loop in and/or out; otherwise only fade-in first cycle.</td>
</tr>
<tr>
<td>By default, volume fading is applied to the beginning and end of</td>
</tr>
</tbody>
</table>
the playback range, i.e. a fade-in segment is placed at the beginning of the sound and a fade-out segment is paced at the end of a sound. However, when looping playback, this may be undesirable as each iteration of the sound will then have a fade-in and fade-out effect.

To set up looping sounds such that a fade-in is applied only when the sound is first started (or playback resumed) and a fade-out is only applied when the sound is explicitly paused or stopped, set this field to true.

Default is false.

**Volume Fades**

<table>
<thead>
<tr>
<th><strong>Ease</strong></th>
<th><strong>SFXDescription::fadeOutEase</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Easing curve for fade-out transition.</td>
<td></td>
</tr>
<tr>
<td>Volume fade-outs will interpolate volume along this curve.</td>
<td></td>
</tr>
</tbody>
</table>

**Volume Fades**

<table>
<thead>
<tr>
<th><strong>float</strong></th>
<th><strong>SFXDescription::fadeOutTime</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of seconds to gradually fade out volume down to zero when playback is stopped or paused.</td>
<td></td>
</tr>
<tr>
<td>Must be &gt;=0.</td>
<td></td>
</tr>
</tbody>
</table>

**Volume Fades**

<table>
<thead>
<tr>
<th><strong>bool</strong></th>
<th><strong>SFXDescription::is3D</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>If true, sounds played with this description will have a position and</td>
<td></td>
</tr>
</tbody>
</table>
orientation in space.

Unlike a non-positional sound, a 3D sound will have its volume attenuated depending on the distance to the listener in space. The farther the sound moves away from the listener, the less audible it will be.

Non-positional sounds, in contrast, will remain at their original volume regardless of where the listener is.

**Note:**
Whether a sound is positional or non-positional cannot be changed once the sound was created so this field determines up front which is the case for a given sound.

### 3D Audio Volume Attenuation

<table>
<thead>
<tr>
<th>bool SFXDescription::isLooping</th>
</tr>
</thead>
<tbody>
<tr>
<td>If true, the sound will be played in an endless loop.</td>
</tr>
<tr>
<td>Default is false.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool SFXDescription::isStreaming</th>
</tr>
</thead>
<tbody>
<tr>
<td>If true, incrementally stream sounds; otherwise sounds are loaded in full.</td>
</tr>
</tbody>
</table>

### Streaming vs. Buffered Audio

<table>
<thead>
<tr>
<th>float SFXDescription::maxDistance</th>
</tr>
</thead>
<tbody>
<tr>
<td>The distance at which attenuation stops.</td>
</tr>
<tr>
<td>In the linear distance model, the attenuated volume will be zero at</td>
</tr>
</tbody>
</table>
this distance.

In the logarithmic model, attenuation will simply stop at this distance and the sound will keep its attenuated volume from there on out. As such, it primarily functions as a cutoff factor to exponential distance attenuation to limit the number of voices relevant to updates.

Only applies to 3D sounds.

**See also:**
- LevelInfo::soundDistanceModel

3D Audio Volume Attenuation

**string SFXDescription::parameters[8]**

Names of the parameters to which sources using this description will automatically be linked.

Individual parameters are identified by their `internalName`.

Interactive Audio

**float SFXDescription::pitch**

Pitch shift to apply to playback.

The pitch assigned to a sound determines the speed at which it is played back. A pitch shift of 1 plays the sound at its default speed. A greater shift factor speeds up playback and a smaller shift factor slows it down.

Must be >0. Default is 1.
**float SFXDescription::priority**

Priority level for virtualization of sounds (1 = base level).

When there are more concurrently active sounds than supported by the audio mixer, some of the sounds need to be culled. Which sounds are culled first depends primarily on total audibility of individual sounds. However, the priority of individual sounds may be decreased or decreased through this field.

---

**Sounds and Voices**

**float SFXDescription::referenceDistance**

Distance at which volume attenuation begins.

Up to this distance, the sound retains its base volume.

In the linear distance model, the volume will linearly from this distance onwards up to maxDistance where it reaches zero.

In the logarithmic distance model, the reference distance determine how fast the sound volume decreases with distance. Each referenceDistance steps (scaled by the rolloff factor), the volume halves.

A rule of thumb is that for sounds that require you to be close to hear them in the real world, set the reference distance to small values whereas for sounds that are widely audible set it to larger values.

Only applies to 3D sounds.

**See also:**

LevellInfo::soundDistanceModel

3D Audio Volume Attenuation
const int SFXDescription::REVERB_DIRECTHFAUTO [static]

Automatic setting of SFXDescription::reverbDirect due to distance to listener.

See also:
SFXDescription::flags

const int SFXDescription::REVERB_INSTANCE0 [static]

EAX4/SFX/GameCube/Wii: Specify channel to target reverb instance 0. Default target.

See also:
SFXDescription::flags

const int SFXDescription::REVERB_INSTANCE1 [static]

EAX4/SFX/GameCube/Wii: Specify channel to target reverb instance 1.

See also:
SFXDescription::flags

const int SFXDescription::REVERB_INSTANCE2 [static]

EAX4/SFX/GameCube/Wii: Specify channel to target reverb instance 2.

See also:
SFXDescription::flags
**const int SFXDescription::REVERB_INSTANCE3** [static]

EAX4/SFX/GameCube/Wii: Specify channel to target reverb instance 3.

**See also:**

SFXDescription::flags

**const int SFXDescription::REVERB_ROOMAUTO** [static]

Automatic setting of SFXDescription::reverbRoom due to distance to listener.

**See also:**

SFXDescription::flags

**const int SFXDescription::REVERB_ROOMHFAUTO** [static]

Automatic setting of SFXDescription::reverbRoomHF due to distance to listener.

**See also:**

SFXDescription::flags

**float SFXDescription::reverbAirAbsorptionFactor**

Multiplies SFXEnvironment::airAbsorptionHR.

**Note:**

SUPPORTED: EAX Only

**See also:**

**int SFXDescription::reverbDirect**

Direct path level (at low and mid frequencies).

**Note:**

SUPPORTED: EAX/I3DL2/FMODSFX

**See also:**


**int SFXDescription::reverbDirectHF**

Relative direct path level at high frequencies.

**Note:**

SUPPORTED: EAX/I3DL2

**See also:**


**float SFXDescription::reverbDopplerFactor**

Per-source doppler factor.

**Note:**

SUPPORTED: EAX Only

**See also:**


**int SFXDescription::reverbExclusion**
Main exclusion control (attenuation at high frequencies).

**Note:**

SUPPORTED: EAX Only

**See also:**


```c
float SFXDescription::reverbExclusionLFRatio
```

Exclusion low-frequency level re. main control.

**Note:**

SUPPORTED: EAX Only

**See also:**


```c
int SFXDescription::reverbFlags
```

Bitfield combination of per-sound reverb flags.

**See also:**

- REVERB_DIRECTHFAUTO
- REVERB_ROOMAUTO
- REVERB_ROOMHFAUTO
- REVERB_INSTANCE0
- REVERB_INSTANCE1
- REVERB_INSTANCE2
- REVERB_INSTANCE3

```c
int SFXDescription::reverbObstruction
```
Main obstruction control (attenuation at high frequencies).

**Note:**
SUPPORTED: EAX/I3DL2

**See also:**

```cpp
float SFXDescription::reverbObstructionLFRatio
```

Obstruction low-frequency level re. main control.

**Note:**
SUPPORTED: EAX/I3DL2

**See also:**

```cpp
int SFXDescription::reverbOcclusion
```

Main occlusion control (attenuation at high frequencies).

**Note:**
SUPPORTED: EAX/I3DL2

**See also:**

```cpp
float SFXDescription::reverbOcclusionDirectRatio
```

Relative occlusion control for direct path.
**Note:**
SUPPORTED: EAX Only

**See also:**

```cpp
float SFXDescription::reverbOcclusionLFRatio
```

Occlusion low-frequency level re. main control.

**Note:**
SUPPORTED: EAX/I3DL2

**See also:**

```cpp
float SFXDescription::reverbOcclusionRoomRatio
```

Relative occlusion control for room effect.

**Note:**
SUPPORTED: EAX Only

**See also:**

```cpp
int SFXDescription::reverbOutsideVolumeHF
```

Outside sound cone level at high frequencies.

**Note:**
SUPPORTED: EAX Only
See also:

float SFXDescription::reverbReverbRolloffFactor

Per-source logarithmic falloff factor.

Note:
SUPPORTED: EAX Only

See also:

int SFXDescription::reverbRoom

Room effect level (at low and mid frequencies).

Note:
SUPPORTED: EAX/I3DL2/FMODSFX

See also:

int SFXDescription::reverbRoomHF

Relative room effect level at high frequencies.

Note:
SUPPORTED: EAX/I3DL2

See also:
**float SFXDescription::reverbRoomRolloffFactor**

Room effect falloff factor.

**Note:**

SUPPORTED: EAX/I3DL2

**See also:**


---

**float SFXDescription::rolloffFactor**

Scale factor to apply to logarithmic distance attenuation curve. If -1, the global rolloff setting is used.

**Note:**

Per-sound rolloff is only supported on OpenAL and FMOD at the moment. With other devices, the global rolloff setting is used for all sounds.

**See also:**

LevelInfo::soundDistanceModel

---

**Point3F SFXDescription::scatterDistance**

Bounds on random displacement of 3D sound positions.

When a 3D sound is created and given its initial position in space, this field is used to determine the amount of randomization applied to the actual position given to the sound system.

The randomization uses the following scheme:

```
x += rand( - scatterDistance[ 0 ], scatterDis...```
\[
y += \text{rand}( - \text{scatterDistance}[1], \text{scatterDistance}[1])
\]
\[
z += \text{rand}( - \text{scatterDistance}[2], \text{scatterDistance}[2])
\]

**SFXSource SFXDescription::sourceGroup**

Group that sources playing with this description should be put into.

When a sound source is allocated, it will be made a child of the source group that is listed in its description. This group will then modulate several properties of the sound as it is played.

For example, one use of groups is to segregate sounds so that volume levels of different sound groups such as interface audio and game audio can be controlled independently.

**Source Hierarchies**

**int SFXDescription::streamPacketSize**

Number of seconds of sample data per single streaming packet.

This field allows to fine-tune streaming for individual sounds. The streaming system processes streamed sounds in batches called packets. Each packet will contain a set amount of sample data determined by this field. The greater its value, the more sample data each packet contains, the more work is done per packet.

**Note:**

This field only takes effect when Torque's own sound system performs the streaming. When FMOD is used, this field is ignored and streaming is performed by FMOD.

**Streaming vs. Buffered Audio**

**int SFXDescription::streamReadAhead**
Number of sample packets to read and buffer in advance.

This field determines the number of packets that the streaming system will try to keep buffered in advance. As such it determines the number of packets that can be consumed by the sound device before the playback queue is running dry. Greater values thus allow for more lag in the streaming pipeline.

**Note:**

This field only takes effect when Torque's own sound system performs the streaming. When FMOD is used, this field is ignored and streaming is performed by FMOD.

**Streaming vs. Buffered Audio**

<table>
<thead>
<tr>
<th>bool SFXDescription::useCustomReverb</th>
</tr>
</thead>
</table>

If true, use the reverb properties defined here on sounds.

By default, sounds will be assigned a generic reverb profile. By setting this flag to true, a custom reverb setup can be defined using the "Reverb" properties that will then be assigned to sounds playing with the description.

**Audio Reverb**

<table>
<thead>
<tr>
<th>bool SFXDescription::useHardware</th>
</tr>
</thead>
</table>

Whether the sound is allowed to be mixed in hardware.

If true, the sound system will try to allocate the voice for the sound directly on the sound hardware for mixing by the hardware mixer. Be aware that a hardware mixer may not provide all features available to sounds mixed in software.

**Note:**
This flag currently only takes effect when using FMOD. Generally, it is preferable to let sounds be mixed in software.

<table>
<thead>
<tr>
<th>float SFXDescription::volume</th>
</tr>
</thead>
</table>

Base volume level for the sound.

This will be the starting point for volume attenuation on the sound. The final effective volume of a sound will be dependent on a number of parameters.

Must be between 0 (mute) and 1 (full volume). Default is 1.

**Volume Attenuation**

Copyright © GarageGames, LLC. All Rights Reserved.
SFXEmitter Class Reference
[Sound]

An invisible 3D object that emits sound. More...

Inheritance diagram for SFXEmitter:

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Class</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SFXSource</td>
<td>getSource()</td>
<td>Get the sound source object from the emitter.</td>
</tr>
<tr>
<td>void</td>
<td>play()</td>
<td>Manually start playback of the emitter's sound.</td>
</tr>
<tr>
<td>void</td>
<td>stop()</td>
<td>Manually stop playback of the emitter's sound.</td>
</tr>
</tbody>
</table>
## Public Attributes

### 3D Sound

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>coneInsideAngle</td>
<td>Angle of inner volume cone of 3D sound in degrees.</td>
</tr>
<tr>
<td>int</td>
<td>coneOutsideAngle</td>
<td>Angle of outer volume cone of 3D sound in degrees.</td>
</tr>
<tr>
<td>float</td>
<td>coneOutsideVolume</td>
<td>Volume scale factor of outside of outer volume 3D sound cone.</td>
</tr>
<tr>
<td>bool</td>
<td>is3D</td>
<td>Whether to play <code>fileName</code> as a positional (3D) sound or not.</td>
</tr>
<tr>
<td>float</td>
<td>maxDistance</td>
<td>Distance at which to stop volume attenuation of the 3D sound.</td>
</tr>
<tr>
<td>float</td>
<td>referenceDistance</td>
<td>Distance at which to start volume attenuation of the 3D sound.</td>
</tr>
<tr>
<td>Point3F</td>
<td>scatterDistance</td>
<td>Bounds on random offset to apply to initial 3D sound position.</td>
</tr>
</tbody>
</table>

### Sound

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>fadeInTime</td>
<td>Number of seconds to gradually fade in volume from zero when playback starts.</td>
</tr>
<tr>
<td>float</td>
<td>fadeOutTime</td>
<td>Number of seconds to gradually fade out volume down to zero when playback is stopped or paused.</td>
</tr>
<tr>
<td><strong>bool</strong> <code>isLooping</code></td>
<td>Whether to play <code>fileName</code> in an infinite loop.</td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>bool</strong> <code>isStreaming</code></td>
<td>Whether to use streamed playback for <code>fileName</code>.</td>
<td></td>
</tr>
<tr>
<td><strong>float</strong> <code>pitch</code></td>
<td>Pitch shift to apply to the sound. Default is 1 = play at normal speed.</td>
<td></td>
</tr>
<tr>
<td><strong>bool</strong> <code>playOnAdd</code></td>
<td>Whether playback of the emitter's sound should start as soon as the emitter object is added to the level.</td>
<td></td>
</tr>
<tr>
<td><strong>SFXSource</strong> <code>sourceGroup</code></td>
<td>The <code>SFXSource</code> to which to assign the sound of this emitter as a child.</td>
<td></td>
</tr>
<tr>
<td><strong>bool</strong> <code>useTrackDescriptionOnly</code></td>
<td>If this is true, all fields except for <code>playOnAdd</code> and <code>track</code> are ignored on the emitter object.</td>
<td></td>
</tr>
<tr>
<td><strong>float</strong> <code>volume</code></td>
<td>Volume level to apply to the sound.</td>
<td></td>
</tr>
</tbody>
</table>

**Media**

<table>
<thead>
<tr>
<th><strong>filename</strong> <code>fileName</code></th>
<th>The sound file to play.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SFXTrack</strong> <code>track</code></td>
<td>The track which the emitter should play.</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Enables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Enables selection of all instances of this type.</td>
</tr>
<tr>
<td>static Color</td>
<td>renderColorInnerCone</td>
<td>The color with which to render dots in the inner sound cone (Editor only).</td>
</tr>
<tr>
<td>static Color</td>
<td>renderColorOuterCone</td>
<td>The color with which to render dots in the outer sound cone (Editor only).</td>
</tr>
<tr>
<td>static Color</td>
<td>renderColorOutsideVolume</td>
<td>The color with which to render dots outside of the outer sound cone (Editor only).</td>
</tr>
<tr>
<td>static Color</td>
<td>renderColorPlayingInRange</td>
<td>The color with which to render a sound emitter's marker cube in the editor when the emitter's sound is playing and in range of the listener.</td>
</tr>
<tr>
<td>static Color</td>
<td>renderColorPlayingOutOfRange</td>
<td>The color with which to render a sound emitter's marker cube in the editor when the emitter's sound is playing but out of the range of the listener.</td>
</tr>
<tr>
<td>static Color</td>
<td>renderColorRangeSphere</td>
<td>The color of the range sphere with which to render sound emitters in the editor.</td>
</tr>
<tr>
<td>static Color</td>
<td>renderColorStoppedInRange</td>
<td>The color with which to render a sound emitter's marker cube in the editor when the emitter's sound is not playing but the emitter is in range of the listener.</td>
</tr>
<tr>
<td>static Color</td>
<td>renderColorStoppedOutOfRange</td>
<td>The color with which to render a sound emitter's</td>
</tr>
</tbody>
</table>
marker cube in the editor when the emitter’s sound is not playing and the emitter is out of range of the listener.

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>renderEmitters</td>
<td>Whether to render enhanced range feedback in the editor on all emitters regardless of selection state.</td>
</tr>
<tr>
<td>static float</td>
<td>renderPointDistance</td>
<td>The distance between individual points in the sound emitter rendering in the editor as the points move from the emitter's center away to maxDistance.</td>
</tr>
<tr>
<td>static float</td>
<td>renderRadialIncrements</td>
<td>The stepping (in degrees) for the radial sweep along the axis of the XY plane sweep for sound emitter rendering in the editor.</td>
</tr>
<tr>
<td>static float</td>
<td>renderSweepIncrements</td>
<td>The stepping (in degrees) for the radial sweep on the XY plane for sound emitter rendering in the editor.</td>
</tr>
</tbody>
</table>
**Detailed Description**

An invisible 3D object that emits sound.

Sound emitters are used to place sounds in the level. They are full 3D objects with their own position and orientation and when assigned 3D sounds, the transform and velocity of the sound emitter object will be applied to the 3D sound.

Sound emitters can be set up of in either of two ways:

- By assigning an existing **SFXTrack** to the emitter's **track** property.

  In this case the general sound setup (3D, streaming, looping, etc.) will be taken from **track**. However, the emitter's own properties will still override their corresponding properties in the track's **SFXDescription**.

- By directly assigning a sound file to the emitter's **fileName** property.

  In this case, the sound file will be set up for playback according to the properties defined on the emitter.

Using **playOnAdd** emitters can be configured to start playing immediately when they are added to the system (e.g. when the level objects are loaded from the mission file).

**Note:**

A sound emitter need not necessarily emit a 3D sound. Instead, sound emitters may also be used to play non-positional sounds. For placing background audio to a level, however, it is usually easier to use **LevelInfo::soundAmbience**.
Sound Emitters and Networking

It is important to be aware of the fact that sounds will only play client-side whereas SFXEmitter objects are server-side entities. This means that a server-side object has no connection to the actual sound playing on the client. It is thus not possible for the server-object to perform queries about playback status and other source-related properties as these may in fact differ from client to client.
Member Function Documentation

**SFXSource SFXEmitter::getSource ( )**

Get the sound source object from the emitter.

**Returns:**

The sound source used by the emitter or null.

**Note:**

This method will return null when called on the server-side SFXEmitter object. Only client-side ghosts actually hold on to SFXSources.

**void SFXEmitter::play ( )**

Manually start playback of the emitter's sound.

If this is called on the server-side object, the play command will be related to all client-side ghosts.

**void SFXEmitter::stop ( )**

Manually stop playback of the emitter's sound.

If this is called on the server-side object, the stop command will be related to all client-side ghosts.
**Member Data Documentation**

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Member Name</th>
<th>Description</th>
<th>Note</th>
<th>See also</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><code>SFXEmitter::coneInsideAngle</code></td>
<td>Angle of inner volume cone of 3D sound in degrees.</td>
<td>This field is ignored if <code>useTrackDescriptionOnly</code> is true.</td>
<td><code>SFXDescription::coneInsideAngle</code></td>
</tr>
<tr>
<td>int</td>
<td><code>SFXEmitter::coneOutsideAngle</code></td>
<td>Angle of outer volume cone of 3D sound in degrees.</td>
<td>This field is ignored if <code>useTrackDescriptionOnly</code> is true.</td>
<td><code>SFXDescription::coneOutsideAngle</code></td>
</tr>
<tr>
<td>float</td>
<td><code>SFXEmitter::coneOutsideVolume</code></td>
<td>Volume scale factor of outside of outer volume 3D sound cone.</td>
<td>This field is ignored if <code>useTrackDescriptionOnly</code> is true.</td>
<td><code>SFXDescription::coneOutsideVolume</code></td>
</tr>
</tbody>
</table>
**float SFXEmitter::fadeInTime**

Number of seconds to gradually fade in volume from zero when playback starts.

**Note:**

This field is ignored if `useTrackDescriptionOnly` is true.

**See also:**

*SFXDescription::fadeInTime*

---

**float SFXEmitter::fadeOutTime**

Number of seconds to gradually fade out volume down to zero when playback is stopped or paused.

**Note:**

This field is ignored if `useTrackDescriptionOnly` is true.

**See also:**

*SFXDescription::fadeOutTime*

---

**filename SFXEmitter::fileName**

The sound file to play.

Use **either** this property or **track**. If both are assigned, **track** takes precedence. The primary purpose of this field is to avoid the need for the user to define **SFXTrack** datablocks for all sounds used in a level.

---

**bool SFXEmitter::is3D**
Whether to play `fileName` as a positional (3D) sound or not.

If a `track` is assigned, the value of this field is ignored.

See also:

- `SFXDescription::is3D`

`bool SFXEmitter::isLooping`

Whether to play `fileName` in an infinite loop.

If a `track` is assigned, the value of this field is ignored.

See also:

- `SFXDescription::isLooping`

`bool SFXEmitter::isStreaming`

Whether to use streamed playback for `fileName`.

If a `track` is assigned, the value of this field is ignored.

See also:

- `SFXDescription::isStreaming`

**Streaming vs. Buffered Audio**

`float SFXEmitter::maxDistance`

Distance at which to stop volume attenuation of the 3D sound.

**Note:**

This field is ignored if `useTrackDescriptionOnly` is true.
See also:

SFXDescription::maxDistance

float SFXEmitter::pitch

Pitch shift to apply to the sound. Default is 1 = play at normal speed.

Note:
This field is ignored if useTrackDescriptionOnly is true.

See also:

SFXDescription::pitch

bool SFXEmitter::playOnAdd

Whether playback of the emitter's sound should start as soon as the emitter object is added to the level.

If this is true, the emitter will immediately start to play when the level is loaded.

float SFXEmitter::referenceDistance

Distance at which to start volume attenuation of the 3D sound.

Note:
This field is ignored if useTrackDescriptionOnly is true.

See also:

SFXDescription::referenceDistance
### Point3F SFXEmitter::scatterDistance

Bounds on random offset to apply to initial 3D sound position.

**Note:**
This field is ignored if `useTrackDescriptionOnly` is true.

**See also:**
- `SFXDescription::scatterDistance`

### SFXSource SFXEmitter::sourceGroup

The `SFXSource` to which to assign the sound of this emitter as a child.

**Note:**
This field is ignored if `useTrackDescriptionOnly` is true.

**See also:**
- `SFXDescription::sourceGroup`

### SFXTrack SFXEmitter::track

The track which the emitter should play.

**Note:**
If assigned, this field will take precedence over a `fileName` that may also be assigned to the emitter.

### bool SFXEmitter::useTrackDescriptionOnly

If this is true, all fields except for `playOnAdd` and `track` are ignored on the emitter object.
This is useful to prevent fields in the track's description from being overridden by emitter fields.

**float SFXEmitter::volume**

Volume level to apply to the sound.

**Note:**
This field is ignored if useTrackDescriptionOnly is true.

**See also:**
SFXDescription::volume
SFXEnvironment Class Reference
[Sound]

Description of a reverb environment. More...

Inheritance diagram for SFXEnvironment:

List of all members.
**Public Attributes**

**Reverb**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float airAbsorptionHF</td>
<td>Change in level per meter at high frequencies.</td>
</tr>
<tr>
<td>float decayHFRatio</td>
<td>High-frequency to mid-frequency decay time ratio.</td>
</tr>
<tr>
<td>float decayLFRatio</td>
<td>Low-frequency to mid-frequency decay time ratio.</td>
</tr>
<tr>
<td>float decayTime</td>
<td>Reverberation decay time at mid frequencies.</td>
</tr>
<tr>
<td>float density</td>
<td>Value that controls the modal density in the late reverberation decay.</td>
</tr>
<tr>
<td>float diffusion</td>
<td>Value that controls the echo density in the late reverberation decay.</td>
</tr>
<tr>
<td>float echoDepth</td>
<td>Echo depth.</td>
</tr>
<tr>
<td>float echoTime</td>
<td>Echo time.</td>
</tr>
<tr>
<td>float envDiffusion</td>
<td>Environment diffusion.</td>
</tr>
<tr>
<td>float envSize</td>
<td>Environment size in meters.</td>
</tr>
<tr>
<td>int flags</td>
<td>A bitfield of reverb flags.</td>
</tr>
<tr>
<td>float HFReference</td>
<td>Reference high frequency in Hertz.</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>float LFReference</td>
<td>Reference low frequency in Hertz.</td>
</tr>
<tr>
<td>float modulationDepth</td>
<td>Modulation depth.</td>
</tr>
<tr>
<td>float modulationTime</td>
<td>Modulation time.</td>
</tr>
<tr>
<td>int reflections</td>
<td>Early reflections level relative to room effect.</td>
</tr>
<tr>
<td>float reflectionsDelay</td>
<td>Initial reflection delay time.</td>
</tr>
<tr>
<td>int reverb</td>
<td>Late reverberation level relative to room effect.</td>
</tr>
<tr>
<td>float reverbDelay</td>
<td>Late reverberation delay time relative to initial reflection.</td>
</tr>
<tr>
<td>float reverbPan [3]</td>
<td>Late reverberation panning vector.</td>
</tr>
<tr>
<td>int room</td>
<td>Room effect level at mid-frequencies.</td>
</tr>
<tr>
<td>int roomHF</td>
<td>Relative room effect level at high frequencies.</td>
</tr>
<tr>
<td>int roomLF</td>
<td>Relative room effect level at low frequencies.</td>
</tr>
<tr>
<td>float roomRolloffFactor</td>
<td>Logarithmic distance attenuation rolloff scale factor for reverb room size effect.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Attribute Type</th>
<th>Attribute Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static const int</td>
<td>REVERB_CORE0</td>
<td>PS2 Only - Reverb is applied to CORE0 (hw voices 0-23).</td>
</tr>
<tr>
<td>static const int</td>
<td>REVERB_CORE1</td>
<td>PS2 Only - Reverb is applied to CORE1 (hw voices 24-47).</td>
</tr>
<tr>
<td>static const int</td>
<td>REVERB_DECAYHFLIMIT</td>
<td>SFXEnvironment::airAbsorptionHF affects SFXEnvironment::decayHFRatio.</td>
</tr>
<tr>
<td>static const int</td>
<td>REVERB_DECAYTIMESCALE</td>
<td>SFXEnvironment::envSize affects reverberation decay time.</td>
</tr>
<tr>
<td>static const int</td>
<td>REVERB_ECHOTIMESCALE</td>
<td>SFXEnvironment::envSize affects echo time.</td>
</tr>
<tr>
<td>static const int</td>
<td>REVERB_HIGHQUALITYDPL2REVERB</td>
<td>GameCube/Wii Only - Use high-quality DPL2 reverb.</td>
</tr>
<tr>
<td>static const int</td>
<td>REVERB_HIGHQUALITYREVERB</td>
<td>GameCube/Wii Only - Use high-quality reverb.</td>
</tr>
<tr>
<td>static const int</td>
<td>REVERB_MODULATIONTIMESCALE</td>
<td>SFXEnvironment::envSize affects modulation time.</td>
</tr>
<tr>
<td>static const int</td>
<td>REVERB_REFLECTIONSDELYSCALE</td>
<td>SFXEnvironment::envSize affects initial reflection delay time.</td>
</tr>
<tr>
<td>static const int</td>
<td>REVERB_REFLECTIONSSCALE</td>
<td>SFXEnvironment::envSize affects reflection level.</td>
</tr>
<tr>
<td>static const int</td>
<td>REVERB_REVERBDELAYSCALE</td>
<td>SFXEnvironment::envSize affects late reverberation delay time.</td>
</tr>
<tr>
<td>static const int</td>
<td>REVERB_REVERBSSCALE</td>
<td></td>
</tr>
</tbody>
</table>
SFXEnvironment::envSize affects reflections level.
**Detailed Description**

Description of a reverb environment.

A reverb environment specifies how the audio mixer should render advanced environmental audio effects.

To use reverb environments in your level, set up one or more ambient audio spaces, assign reverb environments appropriately, and then attach the SFXAmbiences to your LevelInfo (taking effect globally) or Zone objects (taking effect locally).

To define your own custom reverb environments, it is usually easiest to adapt one of the pre-existing reverb definitions:

```cpp
singleton SFXEnvironment( AudioEnvCustomUnderwater ) {
    // Override select properties from AudioEnv
};
```

In the Datablock Editor, this can be done by selecting an existing environment to copy from when creating the SFXEnvironment datablock.

For a precise description of reverb audio and the properties of this class, please consult the EAX documentation.

All SFXEnvironment instances are automatically added to the global SFXEnvironmentSet.

**See also:**

**Audio Reverb**
### Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>SFXEnvironment::airAbsorptionHF</code></td>
<td>Change in level per meter at high frequencies.</td>
</tr>
<tr>
<td>float</td>
<td><code>SFXEnvironment::decayHFRatio</code></td>
<td>High-frequency to mid-frequency decay time ratio.</td>
</tr>
<tr>
<td>float</td>
<td><code>SFXEnvironment::decayLFRatio</code></td>
<td>Low-frequency to mid-frequency decay time ratio.</td>
</tr>
<tr>
<td>float</td>
<td><code>SFXEnvironment::decayTime</code></td>
<td>Reverberation decay time at mid frequencies.</td>
</tr>
<tr>
<td>float</td>
<td><code>SFXEnvironment::density</code></td>
<td>Value that controls the modal density in the late reverberation decay.</td>
</tr>
<tr>
<td>float</td>
<td><code>SFXEnvironment::diffusion</code></td>
<td>Value that controls the echo density in the late reverberation decay.</td>
</tr>
<tr>
<td>float</td>
<td><code>SFXEnvironment::echoDepth</code></td>
<td></td>
</tr>
</tbody>
</table>
Echo depth.

**float SFXEnvironment::echoTime**

Echo time.

**float SFXEnvironment::envDiffusion**

Environment diffusion.

**float SFXEnvironment::envSize**

Environment size in meters.

**int SFXEnvironment::flags**

A bitfield of reverb flags.

See also:

- REVERB_DECAYTIMESCALE
- REVERB_REFLECTIONSSCALE
- REVERB_REFLECTIONSDELYSCALE
- REVERB_REVERBSSCALE
- REVERB_REVERBDELYSCALE
- REVERB_DECAYHFLIMIT
- REVERB_ECHOTIMESCALE
- REVERB_MODULATIONTIMESCALE
- REVERB_CORE0
- REVERB_CORE1
- REVERB_HIGHQUALITYREVERB
- REVERB_HIGHQUALITYDPL2REVERB
float SFXEnvironment::HFReference
Reference high frequency in Hertz.

float SFXEnvironment::LFReference
Reference low frequency in Hertz.

float SFXEnvironment::modulationDepth
Modulation depth.

float SFXEnvironment::modulationTime
Modulation time.

int SFXEnvironment::reflections
Early reflections level relative to room effect.

float SFXEnvironment::reflectionsDelay
Initial reflection delay time.

float SFXEnvironment::reflectionsPan[3]
Early reflections panning vector.

int SFXEnvironment::reverb
Late reverberation level relative to room effect.

\texttt{const int SFXEnvironment::REVERB\_CORE0 \ [static]\}

PS2 Only - Reverb is applied to CORE0 (hw voices 0-23).

See also:

\texttt{SFXEnvironment::flags}\n
\texttt{const int SFXEnvironment::REVERB\_CORE1 \ [static]\}

PS2 Only - Reverb is applied to CORE1 (hw voices 24-47).

See also:

\texttt{SFXEnvironment::flags}\n
\texttt{const int SFXEnvironment::REVERB\_DECAYHFLIMIT \ [static]\}

\texttt{SFXEnvironment::airAbsorptionHF affects}
\texttt{SFXEnvironment::decayHFRatio.}\n
See also:

\texttt{SFXEnvironment::flags}\n
\texttt{const int SFXEnvironment::REVERB\_DECAYTIMESCALE \ [static]\}

\texttt{SFXEnvironment::envSize affects reverberation decay time.}\n
See also:

\texttt{SFXEnvironment::flags}
const int SFXEnvironment::REVERB_ECHOTIMESCALE [static]

SFXEnvironment::envSize affects echo time.

See also:
   SFXEnvironment::flags

const int SFXEnvironment::REVERB_HIGHQUALITYDPL2REVERB [static]

GameCube/Wii Only - Use high-quality DPL2 reverb.

See also:
   SFXEnvironment::flags

const int SFXEnvironment::REVERB_HIGHQUALITYREVERB [static]

GameCube/Wii Only - Use high-quality reverb.

See also:
   SFXEnvironment::flags

const int SFXEnvironment::REVERB_MODULATIONTIMESCALE [static]

SFXEnvironment::envSize affects modulation time.

See also:
   SFXEnvironment::flags

const int SFXEnvironment::REVERB_REFLECTIONSDELYTIMESCALE [static]

SFXEnvironment::envSize affects initial reflection delay time.
See also:

*SFXEnvironment::flags*

**const int SFXEnvironment::REVERB_REFLECTIONSSCALE** [static]

*SFXEnvironment::envSize* affects reflection level.

See also:

*SFXEnvironment::flags*

**const int SFXEnvironment::REVERB_REVERBDELYSCALE** [static]

*SFXEnvironment::envSize* affects late reverberation delay time.

See also:

*SFXEnvironment::flags*

**const int SFXEnvironment::REVERB_REVERBSSCALE** [static]

*SFXEnvironment::envSize* affects reflections level.

See also:

*SFXEnvironment::flags*

**float SFXEnvironment::reverbDelay**

Late reverberation delay time relative to initial reflection.

**float SFXEnvironment::reverbPan[3]**

Late reverberation panning vector.
<table>
<thead>
<tr>
<th>Variable Type</th>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>SFXEnvironment::room</td>
<td>Room effect level at mid-frequencies.</td>
</tr>
<tr>
<td>int</td>
<td>SFXEnvironment::roomHF</td>
<td>Relative room effect level at high frequencies.</td>
</tr>
<tr>
<td>int</td>
<td>SFXEnvironment::roomLF</td>
<td>Relative room effect level at low frequencies.</td>
</tr>
<tr>
<td>float</td>
<td>SFXEnvironment::roomRolloffFactor</td>
<td>Logarithmic distance attenuation rolloff scale factor for reverb room size effect.</td>
</tr>
</tbody>
</table>
SFXFMODEvent Class Reference

[FMOD]

A playable sound event in an FMOD Designer audio project. More...

Inheritance diagram for SFXFMODEvent:

![Inheritance Diagram]

List of all members.

[legend]
## Public Attributes

**DO NOT MODIFY!!**

<table>
<thead>
<tr>
<th>SFXFMODEventGroup</th>
<th>fmodGroup</th>
<th>DO NOT MODIFY!!</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>fmodName</td>
<td>DO NOT MODIFY!!</td>
</tr>
<tr>
<td>Point2F</td>
<td>fmodParameterRanges [8]</td>
<td>DO NOT MODIFY!!</td>
</tr>
<tr>
<td>float</td>
<td>fmodParameterValues [8]</td>
<td>DO NOT MODIFY!!</td>
</tr>
</tbody>
</table>
Detailed Description

A playable sound event in an FMOD Designer audio project.
## Member Data Documentation

<table>
<thead>
<tr>
<th>SFXFMODEventGroup SFXFMODEvent::fmodGroup</th>
</tr>
</thead>
<tbody>
<tr>
<td>DO NOT MODIFY!!</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string SFXFMODEvent::fmodName</th>
</tr>
</thead>
<tbody>
<tr>
<td>DO NOT MODIFY!!</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Point2F SFXFMODEvent::fmodParameterRanges[8]</th>
</tr>
</thead>
<tbody>
<tr>
<td>DO NOT MODIFY!!</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float SFXFMODEvent::fmodParameterValues[8]</th>
</tr>
</thead>
<tbody>
<tr>
<td>DO NOT MODIFY!!</td>
</tr>
</tbody>
</table>
SFXFMODEventGroup Class Reference  
[FMOD]

A group of events in an imported FMOD Designer project. More...

Inheritance diagram for SFXFMODEventGroup:

![Inheritance Diagram](image)

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void freeData()</code></td>
<td>Release the resource data for this group and its subgroups.</td>
</tr>
<tr>
<td><code>bool isDataLoaded()</code></td>
<td>Test whether the resource data for this group has been loaded.</td>
</tr>
<tr>
<td><code>bool loadData(bool loadStreams=true, bool loadSamples=true)</code></td>
<td>Load the resource data for this group, if it has not already been loaded (either directly or indirectly through a parent group).</td>
</tr>
</tbody>
</table>
## Public Attributes

**DO NOT MODIFY!!**

<table>
<thead>
<tr>
<th>Class</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SFXFMODEventGroup</td>
<td>fmodGroup</td>
<td>DO NOT MODIFY!!</td>
</tr>
<tr>
<td>string</td>
<td>fmodName</td>
<td>DO NOT MODIFY!!</td>
</tr>
<tr>
<td>SFXFMODProject</td>
<td>fmodProject</td>
<td>DO NOT MODIFY!!</td>
</tr>
</tbody>
</table>
**Detailed Description**

A group of events in an imported FMOD Designer project.

**Note:**

Instances of this class
Member Function Documentation

void SFXFMODEventGroup::freeData()

Release the resource data for this group and its subgroups.

See also:
SFXFMODProject_resources

bool SFXFMODEventGroup::isDataLoaded()

Test whether the resource data for this group has been loaded.

Returns:
True if the resource data for this group is currently loaded.

bool SFXFMODEventGroup::loadData(bool loadStreams = true,
                                  bool loadSamples = true)

Load the resource data for this group, if it has not already been loaded (either directly or indirectly through a parent group).

This method works recursively and thus data for direct and indirect child groups to this group will be loaded as well.

Parameters:
loadStreams Whether to open streams.
loadSamples Whether to load sample banks.

Returns:
True if the data has been successfully loaded; false otherwise.
See also:

SFXFMODProject_resources
Member Data Documentation

*SFXFMODEventGroup* SFXFMODEventGroup::fmodGroup

**DO NOT MODIFY!!**

*string* SFXFMODEventGroup::fmodName

**DO NOT MODIFY!!**

*SFXFMODProject* SFXFMODEventGroup::fmodProject

**DO NOT MODIFY!!**

Copyright © GarageGames, LLC. All Rights Reserved.
SFXFMODEventSource Class Reference
[FMOD]

A sound source controller playing an FMOD Designer event (SFXFMODEvent). More...

Inheritance diagram for SFXFMODEventSource:

```
SimObject
  `-> SimSet
    `-> SimGroup
      `-> SFXSource
        `-> SFXFMODEventSource
```

List of all members.
Detailed Description

A sound source controller playing an FMOD Designer event (SFXFMODEvent).

FMOD event sources are internally created by the sound system to play events from imported FMOD Designer projects.

Note:

This class cannot be instantiated directly by the user. Instead, instances of SFXFMODEventSource will be implicitly created by the sound system when playing an SFXFMODEvent.

Copyright © GarageGames, LLC. All Rights Reserved.
SFXFMODProject Class Reference
[FMOD]

An FMOD Designer project loaded into Torque. More...

Inheritance diagram for SFXFMODProject:

```
  +--- SimObject
     
  +--- SimDataBlock
     
  +--- SFXFMODProject

[legend]
```

List of all members.
Public Attributes

**FMOD**

<table>
<thead>
<tr>
<th>filename</th>
<th>fileName</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The compiled .fev file from FMOD Designer.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>filename</th>
<th>mediaPath</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Path to the media files; if unset, defaults to project directory.</td>
</tr>
</tbody>
</table>
Detailed Description

An FMOD Designer project loaded into Torque.
Resource Loading
Member Data Documentation

**filename SFXFMODProject::fileName**

The compiled .fev file from FMOD Designer.

**filename SFXFMODProject::mediaPath**

*Path* to the media files; if unset, defaults to project directory.

Copyright © GarageGames, LLC. All Rights Reserved.
SFXParameter Class Reference
[Sound]

A sound channel value that can be bound to multiple sound sources.

More...

Inheritance diagram for SFXParameter:

```
SimObject

SFXParameter

[legend]
```

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>String</th>
<th><code>getParameterName()</code></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Get the name of the parameter.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th><code>reset()</code></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reset the parameter's value to its default.</td>
</tr>
</tbody>
</table>

## Callbacks

<table>
<thead>
<tr>
<th>void</th>
<th><code>onUpdate()</code></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Called when the sound system triggers an update on the parameter.</td>
</tr>
</tbody>
</table>
## Public Attributes

### Sound

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SFXChannel</td>
<td>channel</td>
<td>Channel that the parameter controls.</td>
</tr>
<tr>
<td>float</td>
<td>defaultValue</td>
<td>Value to which the parameter is initially set.</td>
</tr>
<tr>
<td>string</td>
<td>description</td>
<td>Textual description of the parameter.</td>
</tr>
<tr>
<td>Point2F</td>
<td>range</td>
<td>Permitted range for <code>value</code>.</td>
</tr>
<tr>
<td>float</td>
<td>value</td>
<td>Current value of the audio parameter.</td>
</tr>
</tbody>
</table>
Detailed Description

A sound channel value that can be bound to multiple sound sources.

Parameters are special objects that isolate a specific property that sound sources can have and allows to bind this isolated instance to multiple sound sources such that when the value of the parameter changes, all sources bound to the parameter will have their respective property changed.

Parameters are identified and referenced by their internalName. This means that the SFXDescription::parameters and SFXTrack::parameters fields should contain the internalNames of the SFXParameter objects which should be attached to the SFXSources when they are created. No two SFXParameters should have the same internalName.

All SFXParameter instances are automatically made children of the SFXParameterGroup.

Note:

To simply control the volume and/or pitch levels of a group of sounds, it is easier and more efficient to use a sound source group rather than SFXParameters (see Source Hierarchies). Simply create a SFXSource object representing the group, assign SFXDescription::sourceGroup of the sounds appropriately, and then set the volume and/or pitch level of the group to modulate the respective properties of all children.
Parameter Updates

Parameters are periodically allowed to update their own values. This makes it possible to attach custom logic to a parameter and have individual parameters synchronize their values autonomously. Use the `onUpdate()` callback to attach script code to a parameter update.

Example:

```javascript
new SFXParameter( EngineRPMLevel )
{
    // Set the name by which this parameter is identified.
    internalName = "EngineRPMLevel";

    // Let this parameter control the pitch.
    channel = "Pitch";

    // Start out with unmodified pitch.
    defaultValue = 1;

    // Add a texture description of what this does.
    description = "Engine RPM Level";
}

// Create a description that automatically attaches the engine RPM parameter.
singleton SFXDescription( EngineRPMSound : AudioLoop2D )
{
    parameters[ 0 ] = "EngineRPMLevel";
};

// Create sound sources for the engine.
sfxCreateSource( EngineRPMSound, "art/sound/engine/enginePrimary"
sfxCreateSource( EngineRPMSound, "art/sound/engine/engineSecondary"
// Setting the parameter value will now affect:
EngineRPMLevel.value = 0.5;
EngineRPMLevel.value = 1.5;

Interactive Audio
Member Function Documentation

**String SFXParameter::getParameterName( )**

Get the name of the parameter.

**Returns:**

The parameter name.

**void SFXParameter::onUpdate( )**

Called when the sound system triggers an update on the parameter.

This occurs periodically during system operation.

**void SFXParameter::reset( )**

Reset the parameter's value to its default.

**See also:**

*SFXParameter::defaultValue*
### Member Data Documentation

<table>
<thead>
<tr>
<th><strong>Class</strong></th>
<th><strong>Member</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>SFXChannel</td>
<td>SFXParameter::channel</td>
<td>Channel that the parameter controls. This controls which property of the sources it is attached to the parameter controls.</td>
</tr>
<tr>
<td>float</td>
<td>SFXParameter::defaultValue</td>
<td>Value to which the parameter is initially set. When the parameter is first added to the system, <code>value</code> will be set to <code>defaultValue</code>.</td>
</tr>
<tr>
<td>string</td>
<td>SFXParameter::description</td>
<td>Textual description of the parameter. Primarily for use in the Audio Parameters dialog of the editor to allow for easier identification of parameters.</td>
</tr>
<tr>
<td>Point2F</td>
<td>SFXParameter::range</td>
<td>Permitted range for <code>value</code>. Minimum and maximum allowed value for the parameter. Both inclusive. For all but the User0-3 channels, this property is automatically set up by <code>SFXParameter</code>.</td>
</tr>
</tbody>
</table>
float SFXParameter::value

Current value of the audio parameter.

All attached sources are notified when this value changes.
SFXPlayList Class Reference
[Sound]

A datablock describing a playback pattern of sounds. More...

Inheritance diagram for SFXPlayList:

List of all members.
## Public Attributes

### Sound

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>delayTimeIn</code> [16]</td>
<td>Seconds to wait after moving into slot before <code>transitionIn</code>.</td>
</tr>
<tr>
<td>Point2F</td>
<td><code>delayTimeInVariance</code> [16]</td>
<td>Bounds on randomization of <code>delayTimeIn</code>.</td>
</tr>
<tr>
<td>float</td>
<td><code>delayTimeOut</code> [16]</td>
<td>Seconds to wait before moving out of slot after <code>transitionOut</code>.</td>
</tr>
<tr>
<td>float</td>
<td><code>fadeTimeIn</code> [16]</td>
<td>Seconds to fade sound in (-1 to use the track's own fadeInTime.).</td>
</tr>
<tr>
<td>Point2F</td>
<td><code>fadeTimeInVariance</code> [16]</td>
<td>Bounds on randomization of <code>fadeInTime</code>.</td>
</tr>
<tr>
<td>float</td>
<td><code>fadeTimeOut</code> [16]</td>
<td>Seconds to fade sound out (-1 to use the track's own fadeOutTime.).</td>
</tr>
</tbody>
</table>

**SFXPlayListLoopMode**

- `loopMode`: Behavior when description has looping enabled.

- `maxDistance` [16]
<table>
<thead>
<tr>
<th><strong>maxDistance</strong> to apply to 3D sounds in this slot (&lt;1 to use maxDistance of track's own description).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Point2F</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>int</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>float</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>Point2F</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>SFXPlayListRandomMode</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>float</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>Point2F</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>int</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>SFXPlayListReplayMode</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>SFXState</strong></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
to play.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SFXPlayListStateMode</strong></td>
<td><code>stateMode [16]</code> &lt;br&gt;Behavior when assigned state is deactivated while slot is playing.</td>
</tr>
<tr>
<td><strong>SFXTrack</strong></td>
<td><code>track [16]</code> &lt;br&gt;Track to play in this slot.</td>
</tr>
<tr>
<td><strong>SFXPlayListTransitionMode</strong></td>
<td><code>transitionIn [16]</code> &lt;br&gt;Behavior when moving into this slot.</td>
</tr>
<tr>
<td><strong>SFXPlayListTransitionMode</strong></td>
<td><code>transitionOut [16]</code> &lt;br&gt;Behavior when moving out of this slot.</td>
</tr>
<tr>
<td><strong>float</strong></td>
<td><code>volumeScale [16]</code> &lt;br&gt;Scale factor to apply to volume of sounds played on this list slot.</td>
</tr>
<tr>
<td><strong>Point2F</strong></td>
<td><code>volumeScaleVariance [16]</code> &lt;br&gt;Bounds on randomization of <code>volumeScale</code>.</td>
</tr>
</tbody>
</table>

**Debug**

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>bool</strong></td>
<td><code>trace</code> &lt;br&gt;Enable/disable execution tracing for this playlist (local only).</td>
</tr>
</tbody>
</table>
Detailed Description

A datablock describing a playback pattern of sounds.

Playlists allow to define intricate playback patterns of individual tracks and thus allow the sound system to be easily used for playing multiple sounds in single operations.

As playlists are SFXTracks, they can thus be used anywhere in the engine where sound data can be assigned.

Each playlist can hold a maximum of 16 tracks. Longer playlists may be constructed by cascading lists, i.e. by creating a playlist that references other playlists.

Processing of a single playlist slot progresses in a fixed set of steps that are invariably iterated through for each slot (except the slot is assigned a state and its state is deactivated; in this case, the controller will exit out of the slot directly):

1. delayIn:

   Waits a set amount of time before processing the slot. This is 0 by default and is determined by the delayTimeIn (seconds to wait) and delayTimeInVariance (bounds on randomization) properties.

2. transitionIn:

   Decides what to do before playing the slot. Defaults to None which makes this stage a no-operation. Alternatively, the slot can be configured to wait for playback of other slots to finish (Wait and WaitAll) or to stop playback of other slots (Stop and StopAll). Note that Wait and Stop always refer to the source that was last started by the list.

3. play:

   Finally, the track attached to the slot is played. However, this will only start playback of the track and then immediately move on to the next stage. It will not wait for the track to finish playing.
Note also that depending on the replay setting for the slot, this stage may pick up a source that is already playing on the slot rather than starting a new one.

Several slot properties (fade times, min/max distance, and volume/pitch scale) are used in this stage.

4. **delayOut:**

   Waits a set amount of time before transitioning out of the slot. This works the same as `delayIn` and is set to 0 by default (i.e. no delay).

5. **transitionOut:**

   Decides what to do after playing the slot. This works like `transitionIn`.

This is a key difference to playlists in normal music players where upon reaching a certain slot, the slot will immediately play and the player then wait for playback to finish before moving on to the next slot.

**Note:**

Be aware that time limits set on slot delays are soft limits. The sound system updates sound sources in discrete (and equally system update frequency dependent) intervals which thus determines the granularity at which time-outs can be handled.
Value Randomization

For greater variety, many of the values for individual slots may be given a randomization limit that will trigger a dynamic variance of the specified base value.

Any given field $xyz$ that may be randomized has a corresponding field $xyz\text{Variance}$ which is a two-dimensional vector. The first number specifies the greatest value that may be subtracted from the given base value (i.e. the $xyz$ field) whereas the second number specifies the greatest value that may be added to the base value. Between these two limits, a random number is generated.

The default variance settings of "0 0" will thus not allow to add or subtract anything from the base value and effectively disable randomization.

Randomization is re-evaluated on each cycle through a list.
Playlists and States

A unique aspect of playlists is that they allow their playback to be tied to the changing set of active sound states. This feature enables playlists to basically connect to an extensible state machine that can be leveraged by the game code to signal a multitude of different gameplay states with the audio system then automatically reacting to state transitions.

Playlists react to states in three ways:

- Before a controller starts processing a slot it checks whether the slot is assigned a state. If this is the case, the controller checks whether the particular state is active. If it is not, the entire slot is skipped. If it is, the controller goes on to process the slot.
- If a controller is in one of the delay stages for a slot that has a state assigned and the state is deactivated, the controller will stop the delay and skip any of the remaining processing stages for the slot.
- Once the play stage has been processed for a slot that has a state assigned, the slot's stateMode will determine what happens with the playing sound source if the slot's state is deactivated while the sound is still playing.

A simple example of how to make use of states in combination with playlists would be to set up a playlist for background music that reacts to the mood of the current gameplay situation. For example, during combat, tenser music could play than during normal exploration. To set this up, different SFXStates would represent different moods in the game and the background music playlist would have one slot set up for each such mood. By making use of volume fades and the PauseWhenDeactivated stateMode, smooth transitions between the various audio tracks can be produced.

Example:

```csharp
// Create a play list from two SFXProfiles
%playList = new SFXPlayList()
{
```
// Use a looped description so the list will loop.

description = AudioMusicLoop2D;

track[ 0 ] = Profile1;
track[ 1 ] = Profile2;
}

// Play the list.
sfxPlayOnce( %playList );
**Member Data Documentation**

**float SFXPlayList::delayTimeIn[16]**

Seconds to wait after moving into slot before `transitionIn`.

**Point2F SFXPlayList::delayTimeInVariance[16]**

Bounds on randomization of `delayTimeIn`.

**Value Randomization**

**float SFXPlayList::delayTimeOut[16]**

Seconds to wait before moving out of slot after `transitionOut`.

**Point2F SFXPlayList::delayTimeOutVariance[16]**

Bounds on randomization of `delayTimeOut`.

**Value Randomization**

**float SFXPlayList::fadeTimeIn[16]**

Seconds to fade sound in (-1 to use the track's own fadeInTime.).

**See also:**

SFXDescription::fadeTimeIn

**Point2F SFXPlayList::fadeTimeInVariance[16]**
Bounds on randomization of fadeInTime.

Value Randomization

<table>
<thead>
<tr>
<th>float SFXPlayList::fadeOutTimeOut[16]</th>
</tr>
</thead>
</table>

Seconds to fade sound out (-1 to use the track's own fadeOutTime.).

See also:

SFXDescription::fadeOutTimeOut

<table>
<thead>
<tr>
<th>Point2F SFXPlayList::fadeOutTimeOutVariance[16]</th>
</tr>
</thead>
</table>

Bounds on randomization of fadeOutTime.

Value Randomization

<table>
<thead>
<tr>
<th>SFXPlayListLoopMode SFXPlayList::loopMode</th>
</tr>
</thead>
</table>

Behavior when description has looping enabled.

The loop mode determines whether the list will loop over a single slot or loop over all the entire list of slots being played.

See also:

SFXDescription::isLooping

<table>
<thead>
<tr>
<th>float SFXPlayList::maxDistance[16]</th>
</tr>
</thead>
</table>

maxDistance to apply to 3D sounds in this slot (<1 to use maxDistance of track's own description).
### See also:

*SFXDescription::maxDistance*

---

**Point2F SFXPlayList::maxDistanceVariance[16]**

Bounds on randomization of *maxDistance*.

**Value Randomization**

---

**int SFXPlayList::numSlotsToPlay**

Number of slots to play.

Up to a maximum of 16, this field determines the number of slots that are taken from the list for playback. Only slots that have a valid *track* assigned will be considered for this.

---

**float SFXPlayList::pitchScale[16]**

Scale factor to apply to pitch of sounds played on this list slot.

This value will scale the actual pitch set on the track assigned to the slot, i.e. a value of 0.5 will cause the track to play at half its assigned speed.

---

**Point2F SFXPlayList::pitchScaleVariance[16]**

Bounds on randomization of *pitchScale*.

**Value Randomization**

---

**SFXPlayListRandomMode SFXPlayList::random**
Slot playback order randomization pattern.

By setting this field to something other than "NotRandom" to order in which slots of the playlist are processed can be changed from sequential to a random pattern. This allows to create more varied playback patterns. Defaults to "NotRandom".

float SFXPlayList::referenceDistance[16]

referenceDistance to set for 3D sounds in this slot (<1 to use referenceDistance of track's own description).

See also:
   SFXDescription::referenceDistance

Point2F SFXPlayList::referenceDistanceVariance[16]

Bounds on randomization of referenceDistance.

Value Randomization

int SFXPlayList::repeatCount[16]

Number of times to loop this slot.

SFXPlayListReplayMode SFXPlayList::replay[16]

Behavior when an already playing sound is encountered on this slot from a previous cycle.

Each slot can have an arbitrary number of sounds playing on it from previous cycles. This field determines how SFXController will handle these sources.
<table>
<thead>
<tr>
<th><strong>SFXState</strong> SFXPlayList::state[16]</th>
</tr>
</thead>
<tbody>
<tr>
<td>State that must be active for this slot to play.</td>
</tr>
</tbody>
</table>

**Playlists and States**

<table>
<thead>
<tr>
<th><strong>SFXPlayListStateMode</strong> SFXPlayList::stateMode[16]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Behavior when assigned state is deactivated while slot is playing.</td>
</tr>
</tbody>
</table>

**Playlists and States**

<table>
<thead>
<tr>
<th><strong>bool</strong> SFXPlayList::trace</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable/disable execution tracing for this playlist (local only).</td>
</tr>
</tbody>
</table>

If this is true, SFXControllers attached to the list will automatically run in trace mode.

<table>
<thead>
<tr>
<th><strong>SFXTrack</strong> SFXPlayList::track[16]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Track to play in this slot.</td>
</tr>
</tbody>
</table>

This must be set for the slot to be considered for playback. Other settings for a slot will not take effect except this field is set.

<table>
<thead>
<tr>
<th><strong>SFXPlayListTransitionMode</strong> SFXPlayList::transitionIn[16]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Behavior when moving into this slot.</td>
</tr>
</tbody>
</table>

After the delayIn time has expired (if any), this slot determines what the controller will do before actually playing the slot.
**SFXPlayList::transitionOut**[16]

Behavior when moving out of this slot.

After the detailTimeOut has expired (if any), this slot determines what the controller will do before moving on to the next slot.

**float SFXPlayList::volumeScale**[16]

Scale factor to apply to volume of sounds played on this list slot.

This value will scale the actual volume level set on the track assigned to the slot, i.e. a value of 0.5 will cause the track to play at half-volume.

**Point2F SFXPlayList::volumeScaleVariance**[16]

Bounds on randomization of `volumeScale`.

*Value Randomization*
SFXProfile Class Reference
[Sound]

Encapsulates a single sound file for playback by the sound system.
More...

Inheritance diagram for SFXProfile:

List of all members.
Public Member Functions

float getSoundDuration ()
Return the length of the sound data in seconds.
## Public Attributes

### Sound

<table>
<thead>
<tr>
<th>filename</th>
<th>fileName</th>
<th>Path to the sound file.</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>preload</td>
<td>Whether to preload sound data when the profile is added to system.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Encapsulates a single sound file for playback by the sound system.

*SFXProfile* combines a sound description (*SFXDescription*) with a sound file such that it can be played by the sound system. To be able to play a sound file, the sound system will always require a profile for it to be created. However, several of the SFX functions (*sfxPlayOnce()*, *sfxCreateSource()*) perform this creation internally for convenience using temporary profile objects.

Sound files can be in either OGG or WAV format. However, extended format support is available when using FMOD. See Supported Sound File Formats.
Profile Loading

By default, the sound data referenced by a profile will be loaded when the profile is first played and the data then kept until either the profile is deleted or until the sound device on which the sound data is held is deleted.

This initial loading may incur a small delay when the sound is first played. To avoid this, a profile may be explicitly set to load its sound data immediately when the profile is added to the system. This is done by setting the preload property to true.

Note:

Sounds using streamed playback (SFXDescription::isStreaming) cannot be preloaded and will thus ignore the preload flag.

Example:

datablock SFXProfile( Shore01Snd )
{
    fileName       = "art/sound/Lakeshore_mono_01"
    description    = Shore01Looping3d;
    preload        = true;
};
Member Function Documentation

float SFXProfile::getSoundDuration() 

Return the length of the sound data in seconds.

**Returns:**

The length of the sound data in seconds or 0 if the sound referenced by the profile could not be found.
**Member Data Documentation**

<table>
<thead>
<tr>
<th>filename SFXProfile::fileName</th>
</tr>
</thead>
</table>

Path to the sound file.

If the extension is left out, it will be inferred by the sound system. This allows to easily switch the sound format without having to go through the profiles and change the filenames there, too.

<table>
<thead>
<tr>
<th>bool SFXProfile::preload</th>
</tr>
</thead>
</table>

Whether to preload sound data when the profile is added to system.

**Note:**

This flag is ignored by streamed sounds.

**Profile Loading**

Copyright © GarageGames, LLC. All Rights Reserved.
SFXSound Class Reference
[Sound]

A sound controller that directly plays a single sound file. More...

Inheritance diagram for SFXSound:

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>float getDuration()</code></td>
<td>Get the total play time (in seconds) of the sound data attached to the sound.</td>
</tr>
<tr>
<td><code>float getPosition()</code></td>
<td>Get the current playback position in seconds.</td>
</tr>
<tr>
<td><code>bool isReady()</code></td>
<td>Test whether the sound data associated with the sound has been fully loaded and is ready for playback.</td>
</tr>
<tr>
<td><code>void setPosition(float position)</code></td>
<td>Set the current playback position in seconds.</td>
</tr>
</tbody>
</table>
**Detailed Description**

A sound controller that directly plays a single sound file.

When playing individual audio files, SFXSounds are implicitly created by the sound system.

Each sound source has an associated play cursor that can be queried and explicitly positioned by the user. The cursor is a floating-point value measured in seconds.

For streamed sources, playback may not be continuous in case the streaming queue is interrupted.

**Note:**

This class cannot be instantiated directly by the user but rather is implicitly created by the sound system when `sfxCreateSource()` or `sfxPlayOnce()` is called on a SFXProfile instance.
Sounds and Voices

To actually emit an audible signal, a sound must allocate a resource on the sound device through which the sound data is being played back. This resource is called 'voice'.

As with other types of resources, the availability of these resources may be restricted, i.e. a given sound device will usually only support a fixed number of voices that are playing at the same time. Since, however, there may be arbitrary many SFXSounds instantiated and playing at the same time, this needs to be solved.

See also:

   SFXDescription::priority
**Member Function Documentation**

float SFXSound::getDuration() 

Get the total play time (in seconds) of the sound data attached to the sound.

**Returns:**

**Note:**

Be aware that for looped sounds, this will not return the total playback time of the sound.

---

float SFXSound::getPosition() 

Get the current playback position in seconds.

**Returns:**

The current play cursor offset.

---

bool SFXSound::isReady() 

Test whether the sound data associated with the sound has been fully loaded and is ready for playback.

For streamed sounds, this will be false during playback when the stream queue for the sound is starved and waiting for data. For buffered sounds, only an initial loading phase will potentially cause isReady to return false.

**Returns:**

True if the sound is ready for playback.
Set the current playback position in seconds.

If the source is currently playing, playback will jump to the new position. If playback is stopped or paused, playback will resume at the given position when play() is called.

**Parameters:**

- `position` The new position of the play cursor (in seconds).
SFXSource Class Reference

[Sound]

Playback controller for a sound source. More...

Inheritance diagram for SFXSource:

```
+-------------+            +-------------------+
| SimObject   |            | SFXSource          |
|             |            |                   |
| SimSet      |            |                   |
|             |            |                   |
| SimGroup    |            |                   |
|             |            |                   |
| SFXController |        | SFXFMODEventSource |
|             |            | SFXSound           |
```

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void addMarker(String name, float pos)</code></td>
<td>Add a notification marker called <code>name</code> at <code>pos</code> seconds of playback.</td>
</tr>
<tr>
<td><code>void addParameter(SFXParameter parameter)</code></td>
<td>Attach <code>parameter</code> to the source.</td>
</tr>
<tr>
<td><code>float getAttenuatedVolume()</code></td>
<td>Get the final effective volume level of the source.</td>
</tr>
<tr>
<td><code>float getFadeInTime()</code></td>
<td>Get the fade-in time set on the source.</td>
</tr>
<tr>
<td><code>float getFadeOutTime()</code></td>
<td>Get the fade-out time set on the source.</td>
</tr>
<tr>
<td><code>SFXParameter getParameter(int index)</code></td>
<td>Get the parameter at the given index.</td>
</tr>
<tr>
<td><code>int getParameterCount()</code></td>
<td>Get the number of SFXParameters that are attached to the source.</td>
</tr>
<tr>
<td><code>float getPitch()</code></td>
<td>Get the pitch scale of the source.</td>
</tr>
<tr>
<td><code>SFXStatus getStatus()</code></td>
<td>Get the current playback status.</td>
</tr>
<tr>
<td><code>float getVolume()</code></td>
<td>Get the current base volume level of the source.</td>
</tr>
<tr>
<td><code>bool isPaused()</code></td>
<td>Test whether the source is currently paused.</td>
</tr>
<tr>
<td><code>bool isPlaying()</code></td>
<td>Test whether the source is currently playing.</td>
</tr>
<tr>
<td><code>bool isStopped()</code></td>
<td>Test whether the source is currently stopped.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>void <strong>pause</strong> (float <strong>fadeOutTime</strong>=-1.f)</td>
<td>Pause playback of the source.</td>
</tr>
<tr>
<td>void <strong>removeParameter</strong> (<strong>SFXParameter</strong> parameter)</td>
<td>Detach <em>parameter</em> from the source.</td>
</tr>
<tr>
<td>void <strong>setCone</strong> (float <strong>innerAngle</strong>, float <strong>outerAngle</strong>, float <strong>outsideVolume</strong>)</td>
<td>Set up the 3D volume cone for the source.</td>
</tr>
<tr>
<td>void <strong>setFadeTimes</strong> (float <strong>fadeInTime</strong>, float <strong>fadeOutTime</strong>)</td>
<td>Set the fade time parameters of the source.</td>
</tr>
<tr>
<td>void <strong>setPitch</strong> (float <strong>pitch</strong>)</td>
<td>Set the pitch scale of the source.</td>
</tr>
<tr>
<td>void <strong>setTransform</strong> (<strong>Point3F</strong> position, <strong>Point3F</strong> direction)</td>
<td>Set the position and direction of the source.</td>
</tr>
<tr>
<td>void <strong>setTransform</strong> (<strong>Point3F</strong> position) void <strong>play</strong> (float <strong>fadeInTime</strong>)</td>
<td>Start playback of the source.</td>
</tr>
<tr>
<td>void <strong>setVolume</strong> (float <strong>volume</strong>)</td>
<td>Set the base volume level for the source.</td>
</tr>
<tr>
<td>void <strong>stop</strong> (float <strong>fadeOutTime</strong>=-1.f)</td>
<td>Stop playback of the source.</td>
</tr>
</tbody>
</table>

**Callbacks**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void <strong>onParameterValueChange</strong> (<strong>SFXParameter</strong> parameter)</td>
<td>Called when a parameter attached to the source changes value.</td>
</tr>
<tr>
<td>void <strong>onStatusChange</strong> (<strong>SFXStatus</strong> <strong>newStatus</strong>)</td>
<td>Called when the playback status of the source changes.</td>
</tr>
</tbody>
</table>
## Public Attributes

### Sound

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SFXDescription</td>
<td>description The playback configuration that determines the initial sound properties and setup.</td>
</tr>
<tr>
<td>string</td>
<td>statusCallback Name of function to call when the status of the source changes.</td>
</tr>
</tbody>
</table>
Detailed Description

Playback controller for a sound source.

All sound playback is driven by SFXSources. Each such source represents an independent playback controller that directly or indirectly affects sound output.

While this class itself is instantiable, such an instance will not by itself emit any sound. This is the responsibility of its subclasses. Note, however, that none of these subclasses must be instantiated directly but must instead be instantiated indirectly through the SFX interface.
Play-Once Sources

Often, a sound source need only exist for the duration of the sound it is playing. In this case so-called "play-once" sources simplify the bookkeeping involved by leaving the deletion of sources that have expired their playtime to the sound system.

Play-once sources can be created in either of two ways:

- **sfxPlayOnce()**: Directly create a play-once source from a SFXTrack or file.
- **sfxDeleteWhenStopped()**: Retroactively turn any source into a play-once source that will automatically be deleted when moving into stopped state.

See also:
- sfxPlayOnce
- sfxDeleteWhenStopped
**Source Hierarchies**

Source are arranged into playback hierarchies where a parent source will scale some of the properties of its children and also hand on any play(), pause(), and stop() commands to them. This allows to easily group sounds into logical units that can then be operated on as a whole.

An example of this is the segregation of sounds according to their use in the game. Volume levels of background music, in-game sound effects, and character voices will usually be controlled independently and putting their sounds into different hierarchies allows to achieve that easily.

The source properties that are scaled by parent values are:

- volume,
- pitch, and
- priority

This means that if a parent has a volume of 0.5, the child will play at half the effective volume it would otherwise have.

Additionally, parents affect the playback state of their children:

- A parent that is in stopped state will force all its direct and indirect children into stopped state.
- A parent that is in paused state will force all its direct and indirect children that are playing into paused state. However, children that are in stopped state will not be affected.
- A parent that is in playing state will not affect the playback state of its children.

Each source maintains a state that is wants to be in which may differ from the state that is enforced on it by its parent. If a parent changes its states in a way that allows a child to move into its desired state, the child will do so.

For logically grouping sources, instantiate the SFXSource class
directly and make other sources children to it. A source thus instantiated will not effect any real sound output on its own but will influence the sound output of its direct and indirect children.

**Note:**

Be aware that the property values used to scale child property values are the *effective* values. For example, the value used to scale the volume of a child is the *effective* volume of the parent, i.e. the volume after fades, distance attenuation, etc. has been applied.

**See also:**

`SFXDescription::sourceGroup`
Volume Attenuation

During its lifetime, the volume of a source will be continually updated. This update process always progresses in a fixed set of steps to compute the final effective volume of the source based on the base volume level that was either assigned from the SFXDescription associated with the source (SFXDescription::volume) or manually set by the user. The process of finding a source's final effective volume is called "volume attenuation". The steps involved in attenuating a source's volume are (in order):

Fading

If the source currently has a fade-effect applied, the volume is interpolated along the currently active fade curve.

Modulation

If the source is part of a hierarchy, it's volume is scaled according to the effective volume of its parent.

Distance Attenuation

If the source is a 3D sound source, then the volume is interpolated according to the distance model in effect and current listener position and orientation (see 3D Audio).

See also:

SFXDescription::volume
SFXDescription::is3d
Volume Fades

To ease-in and ease-out playback of a sound, fade effects may be applied to sources. A fade will either go from zero volume to full effective volume (fade-in) or from full effective volume to zero volume (fade-out).

Fading is coupled to the play(), pause(), and stop() methods as well as to loop iterations when SFXDescription::fadeLoops is true for the source. play() and the start of a loop iteration will trigger a fade-in whereas pause(), stop() and the end of loop iterations will trigger fade-outs.

For looping sources, if SFXDescription::fadeLoops is false, only the initial play() will trigger a fade-in and no further fading will be applied to loop iterations.

By default, the fade durations will be governed by the SFXDescription::fadeInTime and SFXDescription::fadeOutTime properties of the SFXDescription attached to the source. However, these may be overridden on a per-source basis by setting fade times explicitly with setFadeTimes(). Additionally, the set values may be overridden for individual play(), pause(), and stop() calls by supplying appropriate fadeInTime/fadeOutTime parameters.

By default, volume will interpolate linearly during fades. However, custom interpolation curves can be assigned through the SFXDescription::fadeInEase and SFXDescription::fadeOutEase properties.

See also:
   SFXDescription::fadeInTime
   SFXDescription::fadeOutTime
   SFXDescription::fadeInEase
   SFXDescription::fadeOutEase
   SFXDescription::fadeLoops
Sound Cones

See also:

  SFXDescription::coneInsideAngle
  SFXDescription::coneOutsideAngle
  SFXDescription::coneOutsideVolume
Doppler Effect

See also:

sfxGetDopplerFactor
sfxSetDopplerFactor
SFXAmbience::dopplerFactor
**Playback Markers**

Playback markers allow to attach notification triggers to specific playback positions. Once the play cursor crosses a position for which a marker is defined, the onMarkerPassed callback will be triggered on the `SFXSource` thus allowing to couple script logic to .

Be aware that the precision with which marker callbacks are triggered are bound by global source update frequency. Thus there may be a delay between the play cursor actually passing a marker position and the callback being triggered.
Member Function Documentation

```cpp
void SFXSource::addMarker(String name, float pos)
```

Add a notification marker called `name` at `pos` seconds of playback.

**Parameters:**

- `name` Symbolic name for the marker that will be passed to the `onMarkerPassed()` callback.

- `pos` Playback position in seconds when the notification should trigger. Note that this is a soft limit and there may be a delay between the play cursor actually passing the position and the callback being triggered.

**Note:**

For looped sounds, the marker will trigger on each iteration.

**Example:**

```cpp
// Create a new source.
$source = sfxCreateSource( AudioMusicLoop2D,

// Assign a class to the source.
$source.class = "BackgroundMusic";

// Add a playback marker at one minute into playback.
$source.addMarker( "first", 60 );

// Define the callback function. This function will be called when the playback position passes the one minute mark.
function BackgroundMusic::onMarkerPassed(
{
    if( %markerName == "first" )
        echo( "Playback has passed the 60 seconds 60 "
```
void SFXSource::addParameter(SFXParameter parameter)

Attach `parameter` to the source.

Once attached, the source will react to value changes of the given `parameter`. Attaching a parameter will also trigger an initial read-out of the parameter's current value.

**Parameters:**

- `parameter` The parameter to attach to the source.

float SFXSource::getAttenuatedVolume()

Get the final effective volume level of the source.

This method returns the volume level as it is after source group volume modulation, fades, and distance-based volume attenuation have been applied to the base volume level.

**Returns:**

- The effective volume of the source.

Volume Attenuation

float SFXSource::getFadeInTime()

Get the fade-in time set on the source.

This will initially be `SFXDescription::fadeInTime`.
Returns:
The fade-in time set on the source in seconds.

See also:
SFXDescription::fadeInTime

Volume Fades

float SFXSource::getFadeOutTime( )

Get the fade-out time set on the source.
This will initially be SFXDescription::fadeOutTime.

Returns:
The fade-out time set on the source in seconds.

See also:
SFXDescription::fadeOutTime

Volume Fades

SFXParameter SFXSource::getParameter(int index )

Get the parameter at the given index.

Parameters:

\[ index \]

Index of the parameter to fetch. Must be \( 0 \leq index \leq \text{getParameterCount}() \).

Returns:
The parameter at the given \( index \) or null if \( index \) is out of range.
### Example:

```cpp
// Print the name of each parameter attached to the source.
%numParams = %source.getParameterCount();
for( %i = 0; %i < %numParams; %i ++ )
    echo( %source.getParameter( %i ).getParameterName() );
```

See also:
- `getParameterCount`

### int SFXSource::getParameterCount( )

Get the number of SFXParameters that are attached to the source.

**Returns:**
- The number of parameters attached to the source.

**Example:**

```cpp
// Print the name of each parameter attached to the source.
%numParams = %source.getParameterCount();
for( %i = 0; %i < %numParams; %i ++ )
    echo( %source.getParameter( %i ).getParameterName() );
```

See also:
- `getParameter`
- `addParameter`

### float SFXSource::getPitch( )

Get the pitch scale of the source.

Pitch determines the playback speed of the source (default: 1).
**Returns:**

The current pitch scale factor of the source.

**See also:**

- `setPitch`
- `SFXDescription::pitch`

---

**SFXStatus SFXSource::getStatus()**

Get the current playback status.

**Returns:**

The current playback status.

---

**float SFXSource::getVolume()**

Get the current base volume level of the source.

This is not the final effective volume that the source is playing at but rather the starting volume level before source group modulation, fades, or distance-based volume attenuation are applied.

**Returns:**

The current base volume level.

**See also:**

- `setVolume`
- `SFXDescription::volume`

---

**Volume Attenuation**

---

**bool SFXSource::isPaused()**
Test whether the source is currently paused.

**Returns:**
True if the source is in paused state, false otherwise.

**See also:**
- `pause`
- `getStatus`
- `SFXStatus`

```cpp
bool SFXSource::isPlaying()
```

Test whether the source is currently playing.

**Returns:**
True if the source is in playing state, false otherwise.

**See also:**
- `play`
- `getStatus`
- `SFXStatus`

```cpp
bool SFXSource::isStopped()
```

Test whether the source is currently stopped.

**Returns:**
True if the source is in stopped state, false otherwise.

**See also:**
- `stop`
- `getStatus`
- `SFXStatus`
void SFXSource::onParameterValueChange(SFXParameter parameter)

Called when a parameter attached to the source changes value.

This callback will be triggered before the value change has actually been applied to the source.

**Parameters:**

*parameter* The parameter that has changed value.

**Note:**

This is also triggered when the parameter is first attached to the source.

---

void SFXSource::onStatusChange(SFXStatus newStatus)

Called when the playback status of the source changes.

**Parameters:**

*newStatus* The new playback status.

---

void SFXSource::pause(float fadeOutTime = -1.f)

Pause playback of the source.

**Parameters:**

*fadeOutTime* Seconds for the sound to fade down to zero volume. If -1, the SFXDescription::fadeOutTime set in the source's associated description is used. Pass 0 to disable a fade-out effect that may be configured on the description. Be aware that if a fade-out effect is used, the source will not immediately to paused state but will rather remain in playing state until the fade-
void SFXSource::removeParameter(SFXParameter parameter)

Detach *parameter* from the source.

Once detached, the source will no longer react to value changes of the given *parameter*.

If the parameter is not attached to the source, the method will do nothing.

**Parameters:**

*parameter* The parameter to detach from the source.

---

void SFXSource::setCone(float innerAngle, float outerAngle, float outsideVolume)

Set up the 3D volume cone for the source.

**Parameters:**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>innerAngle</em></td>
<td>Angle of the inner sound cone in degrees (0 \leq \text{innerAngle} \leq 360). Must be (0 \leq \text{innerAngle} \leq 360).</td>
</tr>
<tr>
<td><em>outerAngle</em></td>
<td>Angle of the outer sound cone in degrees (0 \leq \text{outerAngle} \leq 360). Must be (0 \leq \text{outerAngle} \leq 360).</td>
</tr>
<tr>
<td><em>outsideVolume</em></td>
<td>Volume scale factor outside of outer cone (0 \leq \text{outsideVolume} \leq 1). Must be (0 \leq \text{outsideVolume} \leq 1).</td>
</tr>
</tbody>
</table>

**Note:**

This method has no effect on the source if the source is not
void SFXSource::setFadeTimes(float fadeInTime, floatfadeOutTime)

Set the fade time parameters of the source.

**Parameters:**

- `fadeInTime` The new fade-in time in seconds.
- `fadeOutTime` The new fade-out time in seconds.

**See also:**

SFXDescription::fadeInTime
SFXDescription::fadeOutTime

*Volume Fades*

void SFXSource::setPitch(float pitch)

Set the pitch scale of the source.

Pitch determines the playback speed of the source (default: 1).

**Parameters:**

- `pitch` The new pitch scale factor.

**See also:**

getPitch
SFXDescription::pitch

void SFXSource::setTransform(Point3F position, Point3F direction)
Start playback of the source.

Set the position and orientation of the source's 3D sound.

**Parameters:**

- *position* The new position in world space.
- *direction* The forward vector.

Set the position of the source's 3D sound.

**Parameters:**

- *position* The new position in world space.

**Note:**

This method has no effect if the source is not a 3D source.

If the sound data for the source has not yet been fully loaded, there will be a delay after calling play and playback will start after the data has become available.

**Parameters:**

- *fadeInTime* Seconds for the sound to reach full volume. If -1, the `SFXDescription::fadeInTime` set in the `fadeInTime` source's associated description is used. Pass 0 to disable a fade-in effect that may be configured on the description.

```cpp
void SFXSource::setVolume(float volume )
```

Set the base volume level for the source.

This volume will be the starting point for source group volume modulation, fades, and distance-based volume attenuation.

**Parameters:**
volume The new base volume level for the source. Must be 0>=volume<=1.

See also:
getVolume

Volume Attenuation

```cpp
void SFXSource::stop(float fadeOutTime = -1.f)
```

Stop playback of the source.

**Parameters:**

- **fadeOutTime**
  Seconds for the sound to fade down to zero volume. If -1, the SFXDescription::fadeOutTime set in the source's associated description is used. Pass 0 to disable a fade-out effect that may be configured on the description. Be aware that if a fade-out effect is used, the source will not immediately transition to stopped state but will rather remain in playing state until the fade-out time has expired.
Member Data Documentation

SFXDescription SFXSource::description

The playback configuration that determines the initial sound properties and setup.

Any SFXSource must have an associated SFXDescription.

string SFXSource::statusCallback

Name of function to call when the status of the source changes.

The source that had its status changed is passed as the first argument to the function and the new status of the source is passed as the second argument.
SFXSpace Class Reference
[Sound]

A volume in space that defines an ambient sound zone. More...

Inheritance diagram for SFXSpace:

```
Legend:
- SimObject
- NetObject
- SceneObject
- SFXSpace
```

List of all members.
## Public Attributes

### Internal

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>edge</td>
<td>For internal use only.</td>
</tr>
<tr>
<td>string</td>
<td>plane</td>
<td>For internal use only.</td>
</tr>
<tr>
<td>string</td>
<td>point</td>
<td>For internal use only.</td>
</tr>
</tbody>
</table>

### Sound

**SFXAmbience**  
**soundAmbience**  
Ambient sound environment for the space.
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

A volume in space that defines an ambient sound zone.
### Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td><strong>SFXSpace::edge</strong></td>
</tr>
<tr>
<td></td>
<td>For internal use only.</td>
</tr>
<tr>
<td>string</td>
<td><strong>SFXSpace::plane</strong></td>
</tr>
<tr>
<td></td>
<td>For internal use only.</td>
</tr>
<tr>
<td>string</td>
<td><strong>SFXSpace::point</strong></td>
</tr>
<tr>
<td></td>
<td>For internal use only.</td>
</tr>
<tr>
<td>SFXAmbience</td>
<td><strong>SFXSpace::soundAmbience</strong></td>
</tr>
<tr>
<td></td>
<td>Ambient sound environment for the space.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
SFXState Class Reference
[Sound]

A boolean switch used to modify playlist behavior. More...

Inheritance diagram for SFXState:

```
  SimObject
     ^
     |        SimDataBlock
     |        ^
     |       |        SFXState
     |       |        [legend]
```

List of all members.
**Public Member Functions**

<table>
<thead>
<tr>
<th>void</th>
<th>activate ()</th>
<th>Increase the activation count on the state.</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>deactivate ()</td>
<td>Decrease the activation count on the state.</td>
</tr>
<tr>
<td>void</td>
<td>disable ()</td>
<td>Increase the disabling count of the state.</td>
</tr>
<tr>
<td>void</td>
<td>enable ()</td>
<td>Decrease the disabling count of the state.</td>
</tr>
<tr>
<td>bool</td>
<td>isActive ()</td>
<td>Test whether the state is currently active.</td>
</tr>
<tr>
<td>bool</td>
<td>isDisabled ()</td>
<td>Test whether the state is currently disabled.</td>
</tr>
</tbody>
</table>

**Callbacks**

<table>
<thead>
<tr>
<th>void</th>
<th>onActivate ()</th>
<th>Called when the state goes from inactive to active.</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>onDeactivate ()</td>
<td>called when the state goes from active to deactivate.</td>
</tr>
</tbody>
</table>
# Public Attributes

**State**

<table>
<thead>
<tr>
<th>SFXState</th>
<th>excludedStates [4]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>States that will automatically be disabled when this state is activated.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SFXState</th>
<th>includedStates [4]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>States that will automatically be activated when this state is activated.</td>
</tr>
</tbody>
</table>
Detailed Description

A boolean switch used to modify playlist behavior.

Sound system states are used to allow playlist controllers to make decisions based on global state. This is useful, for example, to couple audio playback to gameplay state. Certain states may, for example, represent different locations that the listener can be in, like underwater, in open space, or indoors. Other states could represent moods of the current gameplay situation, like, for example, an aggressive mood during combat.

By activating and deactivating sound states according to gameplay state, a set of concurrently running playlists may react and adapt to changes in the game.
**Activation and Deactivation**

At any time, a given state can be either active or inactive. Calling `activate()` on a state increases an internal counter and calling `deactivate()` decreases the counter. Only when the count reaches zero will the state be deactivated.

In addition to the activation count, states also maintain a disabling count. Calling `disable()` increases this count and calling `enable()` decreases it. As long as this count is greater than zero, a given state will not be activated even if its activation count is non-zero. Calling `disable()` on an active state will not only increase the disabling count but also deactivate the state. Calling `enable()` on a state with a positive activation count will re-activate the state when the disabling count reaches zero.
State Dependencies

By listing other states in in its includedStates and excludedStates fields, a state may automatically trigger the activation or disabling of other states in the system when it is activated. This allows to form dependency chains between individual states.

Example:

```cpp
// State indicating that the listener is submerged.
singleton SFXState( AudioLocationUnderwater ) {
    parentGroup = AudioLocation;
    // AudioStateExclusive is a class defined in the core scripts that will automatically
    // ensure for a state to deactivate all siblings in its parentGroup when it is activated.
    className = "AudioStateExclusive";
};

// State suitable e.g. for combat.
singleton SFXState( AudioMoodAggressive ) {
    parentGroup = AudioMood;
    className = "AudioStateExclusive";
};
```

See also:
- SFXPlayList
- SFXController
- SFXPlayList::state
- SFXPlayList::stateMode

Interactive Audio
void SFXState::activate()

Increase the activation count on the state.

If the state isn't already active and it is not disabled, the state will be activated.

See also:
    isActive
deactivate

void SFXState::deactivate()

Decrease the activation count on the state.

If the count reaches zero and the state was not disabled, the state will be deactivated.

See also:
    isActive
    activate

void SFXState::disable()

Increase the disabling count of the state.

If the state is currently active, it will be deactivated.

See also:
    isDisabled
void SFXState::enable()

Decrease the disabling count of the state.

If the disabling count reaches zero while the activation count is still non-zero, the state will be reactivated again.

**See also:**

isDisabled

bool SFXState::isActive()

Test whether the state is currently active.

This is true when the activation count is >0 and the disabling count is =0.

**Returns:**

True if the state is currently active.

**See also:**

activate

bool SFXState::isDisabled()

Test whether the state is currently disabled.

This is true when the disabling count of the state is non-zero.

**Returns:**

True if the state is disabled.

**See also:**

disable
void SFXState::onActivate()

Called when the state goes from inactive to active.

void SFXState::onDeactivate()

called when the state goes from active to deactivate.
Member Data Documentation

**SFXState SFXState::excludedStates[4]**

States that will automatically be disabled when this state is activated.

**Activation and Deactivation**

**SFXState SFXState::includedStates[4]**

States that will automatically be activated when this state is activated.

**Activation and Deactivation**
SFXTrack Class Reference
[Sound]

Abstract base class for sound data that can be played back by the sound system. More...

Inheritance diagram for SFXTrack:

List of all members.
Public Attributes

Sound

<table>
<thead>
<tr>
<th>SFXDescription</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Playback setup description for this track.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>parameters [8]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Parameters to automatically attach to SFXSources created from this track.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Abstract base class for sound data that can be played back by the sound system.

The term "track" is used in the sound system to refer to any entity that can be played back as a sound source. These can be individual files (*SFXProfile*), patterns of other tracks (*SFXPlayList*), or special sound data defined by a device layer (*SFXFMODEvent*).

Any track must be paired with a *SFXDescription* that tells the sound system how to set up playback for the track.

All objects that are of type *SFXTrack* will automatically be added to *SFXTrackSet*.

**Note:**

This class cannot be instantiated directly.
Member Data Documentation

**SFXDescription SFXTrack::description**

Playback setup description for this track.

If unassigned, the description named "AudioEffects" will automatically be assigned to the track. If this description is not defined, track creation will fail.

**string SFXTrack::parameters[8]**

Parameters to automatically attach to SFXSources created from this track.

Individual parameters are identified by their `internalName`. 

Copyright © GarageGames, LLC. All Rights Reserved.
ShaderData Class Reference
[Shaders]

Special type of data block that stores information about a handwritten shader. More...

Inheritance diagram for ShaderData:

List of all members.
Public Member Functions

void reload ()
    Rebuilds all the vertex and pixel shader instances created from this ShaderData.
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>defines</td>
<td>String of case-sensitive defines passed to the shader compiler.</td>
</tr>
<tr>
<td>filename</td>
<td>DXPixelShaderFile</td>
<td>Path to the DirectX pixel shader file to use for this ShaderData.</td>
</tr>
<tr>
<td>filename</td>
<td>DXVertexShaderFile</td>
<td>Path to the DirectX vertex shader file to use for this ShaderData.</td>
</tr>
<tr>
<td>filename</td>
<td>OGLPixelShaderFile</td>
<td>Path to an OpenGL pixel shader file to use for this ShaderData.</td>
</tr>
<tr>
<td>filename</td>
<td>OGLVertexShaderFile</td>
<td>Path to an OpenGL vertex shader file to use for this ShaderData.</td>
</tr>
<tr>
<td>float</td>
<td>pixVersion</td>
<td>Indicates target level the shader should be compiled.</td>
</tr>
<tr>
<td>bool</td>
<td>useDevicePixVersion</td>
<td>If true, the maximum pixel shader version offered by the graphics card will be used.</td>
</tr>
</tbody>
</table>
Detailed Description

Special type of data block that stores information about a handwritten shader.

To use hand written shaders, a ShaderData datablock must be used. This datablock refers only to the vertex and pixel shader filenames and a hardware target level. Shaders are API specific, so DirectX and OpenGL shaders must be explicitly identified.

Example:

```cpp
// Used for the procedural cloud system singleton ShaderData( CloudLayerShader )
{
    DXVertexShaderFile = "shaders/common/cloudLayerV.hlsl",
    DXPixelShaderFile = "shaders/common/cloudLayerP.hlsl",
    OGLVertexShaderFile = "shaders/common/gl/cloudLayerV.glsl",
    OGLPixelShaderFile = "shaders/common/gl/cloudLayerP.glsl",
    pixVersion = 2.0;
};
```
Member Function Documentation

```cpp
void ShaderData::reload()
```

Rebuilds all the vertex and pixel shader instances created from this `ShaderData`.

**Example:**

```cpp
// Rebuild the shader instances from ShaderCloudLayerShader.reload();
```
Member Data Documentation

**string ShaderData::defines**

String of case-sensitive defines passed to the shader compiler.

The string should be delimited by a semicolon, tab, or newline character.

**Example:**

```cpp
singleton ShaderData( FlashShader )
{
    DXVertexShaderFile = "shaders/common/postFx/flashV.hlsl"
    DXPixelShaderFile = "shaders/common/postFx/flashP.hlsl"

    //Define setting the color of WHITE_COLOR.
    defines = "WHITE_COLOR=float4(1.0,1.0,1.0,0.0)"
    pixVersion = 2.0
}
```

**filename ShaderData::DXPixelShaderFile**

Path to the DirectX pixel shader file to use for this ShaderData.

It must contain only one program and no vertex shader, just the pixel shader. It can be either an HLSL or assembly level shader. HLSL's must have a filename extension of .hlsl, otherwise its assumed to be an assembly file.

**filename ShaderData::DXVertexShaderFile**
Path to the DirectX vertex shader file to use for this `ShaderData`.

It must contain only one program and no pixel shader, just the vertex shader. It can be either an HLSL or assembly level shader. HLSL's must have a filename extension of `.hlsl`, otherwise it's assumed to be an assembly file.

<table>
<thead>
<tr>
<th>filename</th>
<th>ShaderData::OGLVertexShaderFile</th>
</tr>
</thead>
</table>

Path to an OpenGL pixel shader file to use for this `ShaderData`.

It must contain only one program and no vertex shader, just the pixel shader.

<table>
<thead>
<tr>
<th>filename</th>
<th>ShaderData::OGLPixelShaderFile</th>
</tr>
</thead>
</table>

Path to an OpenGL vertex shader file to use for this `ShaderData`.

It must contain only one program and no pixel shader, just the vertex shader.

<table>
<thead>
<tr>
<th>float</th>
<th>ShaderData::pixVersion</th>
</tr>
</thead>
</table>

Indicates target level the shader should be compiled.

Valid numbers at the time of this writing are 1.1, 1.4, 2.0, and 3.0. The shader will not run properly if the hardware does not support the level of shader compiled.

<table>
<thead>
<tr>
<th>bool</th>
<th>ShaderData::useDevicePixVersion</th>
</tr>
</thead>
</table>

If true, the maximum pixel shader version offered by the graphics
card will be used.

Otherwise, the script-defined pixel shader version will be used.
ShapeBase Class Reference
[Game Objects]

A scriptable, renderable shape. More...

Inheritance diagram for ShapeBase:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td><code>applyDamage</code> (float amount)</td>
<td>Increment the current damage level by the specified amount.</td>
</tr>
<tr>
<td>bool</td>
<td><code>applyImpulse</code> (Point3F pos, Point3F vec)</td>
<td>Apply an impulse to the object.</td>
</tr>
<tr>
<td>void</td>
<td><code>applyRepair</code> (float amount)</td>
<td>Repair damage by the specified amount.</td>
</tr>
<tr>
<td>bool</td>
<td><code>canCloak</code> ()</td>
<td>Check if this object can cloak.</td>
</tr>
<tr>
<td>void</td>
<td><code>changeMaterial</code> (string mapTo, Material oldMat, Material newMat)</td>
<td>Change one of the materials on the shape.</td>
</tr>
<tr>
<td>bool</td>
<td><code>destroyThread</code> (int slot)</td>
<td>Destroy an animation thread, which prevents it from playing.</td>
</tr>
<tr>
<td>void</td>
<td><code>dumpMeshVisibility</code> ()</td>
<td>Print a list of visible and hidden meshes in the shape to the console for debugging purposes.</td>
</tr>
<tr>
<td>Point3F</td>
<td><code>getAIRepairPoint</code> ()</td>
<td>Get the position at which the AI should stand to repair things.</td>
</tr>
<tr>
<td>float</td>
<td><code>getCameraFov</code> ()</td>
<td>Returns the vertical field of view in degrees for this object if used as a camera.</td>
</tr>
<tr>
<td>int</td>
<td><code>getControllingClient</code> ()</td>
<td>Get the client (if any) that controls this object.</td>
</tr>
<tr>
<td>int</td>
<td><code>getControllingObject</code> ()</td>
<td>Get the object (if any) that controls this object.</td>
</tr>
<tr>
<td>float</td>
<td><code>getDamageFlash</code> ()</td>
<td></td>
</tr>
<tr>
<td>Function Name</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>getDamageLevel()</td>
<td>Get the damage flash level.</td>
<td></td>
</tr>
<tr>
<td>getDamagePercent()</td>
<td>Get the object's current damage level as a percentage of maxDamage.</td>
<td></td>
</tr>
<tr>
<td>getDamageState()</td>
<td>Get the object's damage state.</td>
<td></td>
</tr>
<tr>
<td>getDefaultCameraFov()</td>
<td>Returns the default vertical field of view in degrees for this object if used as a camera.</td>
<td></td>
</tr>
<tr>
<td>getEnergyLevel()</td>
<td>Get the object's current energy level.</td>
<td></td>
</tr>
<tr>
<td>getEnergyPercent()</td>
<td>Get the object's current energy level as a percentage of maxEnergy.</td>
<td></td>
</tr>
<tr>
<td>getEyePoint()</td>
<td>Get the position of the 'eye' for this object.</td>
<td></td>
</tr>
<tr>
<td>getEyeTransform()</td>
<td>Get the 'eye' transform for this object.</td>
<td></td>
</tr>
<tr>
<td>getEyeVector()</td>
<td>Get the forward direction of the 'eye' for this object.</td>
<td></td>
</tr>
<tr>
<td>getImageAltTrigger(int slot)</td>
<td>Get the alt trigger state of the Image mounted in the specified slot.</td>
<td></td>
</tr>
<tr>
<td>getImageAmmo(int slot)</td>
<td>Get the ammo state of the Image mounted in the specified slot.</td>
<td></td>
</tr>
<tr>
<td>getImageGenericTrigger(int slot, int trigger)</td>
<td>Get the generic trigger state of the Image mounted in the specified slot.</td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td><code>bool getImageLoaded (int slot)</code></td>
<td>Get the loaded state of the Image mounted in the specified slot.</td>
<td></td>
</tr>
<tr>
<td><code>string getImageScriptAnimPrefix (int slot)</code></td>
<td>Get the script animation prefix of the Image mounted in the specified slot.</td>
<td></td>
</tr>
<tr>
<td><code>int getImageSkinTag (int slot)</code></td>
<td>Get the skin tag ID for the Image mounted in the specified slot.</td>
<td></td>
</tr>
<tr>
<td><code>string getImageState (int slot)</code></td>
<td>Get the name of the current state of the Image in the specified slot.</td>
<td></td>
</tr>
<tr>
<td><code>bool getImageTarget (int slot)</code></td>
<td>Get the target state of the Image mounted in the specified slot.</td>
<td></td>
</tr>
<tr>
<td><code>bool getImageTrigger (int slot)</code></td>
<td>Get the trigger state of the Image mounted in the specified slot.</td>
<td></td>
</tr>
<tr>
<td><code>string getLookAtPoint (float distance=2000, int typeMask=0xFFFFFFFF)</code></td>
<td>Get the world position this object is looking at.</td>
<td></td>
</tr>
<tr>
<td><code>float getMaxDamage ()</code></td>
<td>Get the object's maxDamage level.</td>
<td></td>
</tr>
<tr>
<td><code>string getModelFile ()</code></td>
<td>Get the model filename used by this shape.</td>
<td></td>
</tr>
<tr>
<td><code>int getMountedImage (int slot)</code></td>
<td>Get the Image mounted in the specified slot.</td>
<td></td>
</tr>
<tr>
<td><code>int getMountSlot (ShapeBaseImageData image)</code></td>
<td>Get the first slot the given datablock is mounted to on this object.</td>
<td></td>
</tr>
<tr>
<td><code>Point3F getMuzzlePoint (int slot)</code></td>
<td>Get the muzzle position of the Image mounted in the</td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>VectorF getMuzzleVector (int slot)</td>
<td>Get the muzzle vector of the Image mounted in the specified slot.</td>
<td></td>
</tr>
<tr>
<td>int getPendingImage (int slot)</td>
<td>Get the Image that will be mounted next in the specified slot.</td>
<td></td>
</tr>
<tr>
<td>float getRechargeRate ()</td>
<td>Get the current recharge rate.</td>
<td></td>
</tr>
<tr>
<td>float getRepairRate ()</td>
<td>Get the per-tick repair amount.</td>
<td></td>
</tr>
<tr>
<td>string getShapename ()</td>
<td>Get the name of the shape.</td>
<td></td>
</tr>
<tr>
<td>string getSkinName ()</td>
<td>Get the name of the skin applied to this shape.</td>
<td></td>
</tr>
<tr>
<td>TransformF getSlotTransform (int slot)</td>
<td>Get the world transform of the specified mount slot.</td>
<td></td>
</tr>
<tr>
<td>int getTargetCount ()</td>
<td>Get the number of materials in the shape.</td>
<td></td>
</tr>
<tr>
<td>string getTargetName (int index)</td>
<td>Get the name of the indexed shape material.</td>
<td></td>
</tr>
<tr>
<td>VectorF getVelocity ()</td>
<td>Get the object's current velocity.</td>
<td></td>
</tr>
<tr>
<td>float getWhiteOut ()</td>
<td>Get the white-out level.</td>
<td></td>
</tr>
<tr>
<td>bool hasImageState (int slot, string state)</td>
<td>Check if the given state exists on the mounted Image.</td>
<td></td>
</tr>
<tr>
<td>bool isCloaked ()</td>
<td>Check if this object is cloaked.</td>
<td></td>
</tr>
<tr>
<td>bool isDestroyed ()</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Method Name</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>bool isDisabled()</td>
<td>Check if the object is in the Disabled or Destroyed damage state.</td>
<td></td>
</tr>
<tr>
<td>bool isEnabled()</td>
<td>Check if the object is in the Enabled damage state.</td>
<td></td>
</tr>
<tr>
<td>bool isHidden()</td>
<td>Check if the object is hidden.</td>
<td></td>
</tr>
<tr>
<td>bool isImageFiring(int slot)</td>
<td>Check if the current Image state is firing.</td>
<td></td>
</tr>
<tr>
<td>bool isImageMounted(ShapeBaseImageData image)</td>
<td>Check if the given datablock is mounted to any slot on this object.</td>
<td></td>
</tr>
<tr>
<td>bool mountImage(ShapeBaseImageData image, int slot, bool loaded=true, string skinTag=&quot;&quot;)</td>
<td>Mount a new Image.</td>
<td></td>
</tr>
<tr>
<td>bool pauseThread(int slot)</td>
<td>Pause an animation thread.</td>
<td></td>
</tr>
<tr>
<td>bool playAudio(int slot, SFXTrack track)</td>
<td>Attach a sound to this shape and start playing it.</td>
<td></td>
</tr>
<tr>
<td>bool playThread(int slot, string name=&quot;&quot;)</td>
<td>Start a new animation thread, or restart one that has been paused or stopped.</td>
<td></td>
</tr>
<tr>
<td>void setAllMeshesHidden(bool hide)</td>
<td>Set the hidden state on all the shape meshes.</td>
<td></td>
</tr>
<tr>
<td>void setCameraFov(float fov)</td>
<td>Set the vertical field of view in degrees for this object if used as a camera.</td>
<td></td>
</tr>
<tr>
<td>void setCloaked(bool cloak)</td>
<td>Set the cloaked state of this object.</td>
<td></td>
</tr>
<tr>
<td>void setDamageFlash(float level)</td>
<td>Set the damage flash level for this object.</td>
<td></td>
</tr>
</tbody>
</table>
Set the damage flash level.

```cpp
void setDamageLevel (float level)
Set the object's current damage level.
```

```cpp
bool setDamageState (string state)
Set the object's damage state.
```

```cpp
void setDamageVector (Point3F vec)
Set the damage direction vector.
```

```cpp
void setEnergyLevel (float level)
Set this object's current energy level.
```

```cpp
void setHidden (bool show)
Add or remove this object from the scene.
```

```cpp
bool setImageAltTrigger (int slot, bool state)
Set the alt trigger state of the Image mounted in the specified slot.
```

```cpp
bool setImageAmmo (int slot, bool state)
Set the ammo state of the Image mounted in the specified slot.
```

```cpp
int setImageGenericTrigger (int slot, int trigger, bool state)
Set the generic trigger state of the Image mounted in the specified slot.
```

```cpp
bool setImageLoaded (int slot, bool state)
Set the loaded state of the Image mounted in the specified slot.
```

```cpp
void setImageScriptAnimPrefix (int slot, string prefix)
Set the script animation prefix for the Image mounted in the specified slot.
```

```cpp
bool setImageTarget (int slot, bool state)
Set the target state of the Image mounted in the specified slot.
```

```cpp
bool setImageTrigger (int slot, bool state)
Set the trigger state of the Image mounted in the specified slot.
```
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void setInvincibleMode (float time, float speed)</td>
<td>Setup the invincible effect.</td>
</tr>
<tr>
<td>void setMeshHidden (string name, bool hide)</td>
<td>Set the hidden state on the named shape mesh.</td>
</tr>
<tr>
<td>void setRechargeRate (float rate)</td>
<td>Set the recharge rate.</td>
</tr>
<tr>
<td>void setRepairRate (float rate)</td>
<td>Set amount to repair damage by each tick.</td>
</tr>
<tr>
<td>void setShapeName (string name)</td>
<td>Set the name of this shape.</td>
</tr>
<tr>
<td>void setSkinName (string name)</td>
<td>Apply a new skin to this shape.</td>
</tr>
<tr>
<td>bool setThreadDir (int slot, bool fwd)</td>
<td>Set the playback direction of an animation thread.</td>
</tr>
<tr>
<td>bool setThreadPosition (int slot, float pos)</td>
<td>Set the position within an animation thread.</td>
</tr>
<tr>
<td>bool setThreadTimeScale (int slot, float scale)</td>
<td>Set the playback time scale of an animation thread.</td>
</tr>
<tr>
<td>bool setVelocity (Point3F vel)</td>
<td>Set the object's velocity.</td>
</tr>
<tr>
<td>void setWhiteOut (float level)</td>
<td>Set the white-out level.</td>
</tr>
<tr>
<td>void startFade (int time, int delay, bool fadeOut)</td>
<td>Fade the object in or out without removing it from the scene.</td>
</tr>
<tr>
<td>bool stopAudio (int slot)</td>
<td>Stop a sound started with playAudio.</td>
</tr>
<tr>
<td>bool stopThread (int slot)</td>
<td>Stop an animation thread.</td>
</tr>
<tr>
<td>bool unmountImage (int slot)</td>
<td></td>
</tr>
</tbody>
</table>
Unmount the mounted Image in the specified slot.

**Callbacks**

| float  validateCameraFov (float fov) |
|-------|----------------------------------|
| Called on the server when the client has requested a FOV change. |
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>isAIControlled</code></td>
<td>Is this object AI controlled.</td>
</tr>
<tr>
<td>string</td>
<td><code>skin</code></td>
<td>The skin applied to the shape.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool <strong>isRenderable</strong></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool <strong>isSelectable</strong></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

A scriptable, renderable shape.

**ShapeBase** is the renderable shape from which most of the scriptable, game objects are derived, including the **Player**, **Vehicle** and **Item** classes. **ShapeBase** provides collision detection, audio channels, and animation as well as damage (and damage states), energy, and the ability to mount Images and objects.

**ShapeBase** objects are not normally instantiated in the scene; derived classes such as **Player**, **WheeledVehicle**, and, **StaticShape** are used instead. But **ShapeBase** (and the associated datablock, **ShapeBaseData**) may be used to provide functionality common to all derived objects.

A **ShapeBase** object consists of a DTS or DAE shape file. This file has the following requirements:

- **Nodes**
  - **cam**
    - This node is used as the 3rd person camera position. If this is not found the 'eye' node is used instead.
  - **eye**
    - This node is used as the 1st person camera position.
  - **ear**
    - This node is where the SFX listener is mounted on. If this is not found the 'eye' node is used instead.
  - **mountN**
    - Nodes used for mounting **ShapeBaseImages** to this object.
      - N is an integer from 0 to 31
  - **AIRepairNode**
    - This node is where AI should stand to repair the object. If not found, the object's origin is used.

- **Sequences Indicating Condition**
visibility
A two frame sequence used to show object damage when the object is destroyed (start=no damage, end=destroyed). This sequence is optional.

damage
Sequence used to show object damage, such as a vehicle getting more dents (start=no damage, end=fully damaged). When the object is fully destroyed this sequence returns to the start and the visibility sequence kicks in. This sequence is optional.

- Detail Levels

collision-N
Convex geometry used for collision detection. N is a integer from 1 to 8. For classes that support polysoup collision, this geometry need not be convex.

LOS-N
Convex geometry used for line-of-sight collision detection. N is an integer from 9 to 16.
Control Object

Generally in a Torque game, each client is in control of a single game object (such as a **Player** in an FPS game, or a **WheeledVehicle** in a racing game). In a game where the client has control over multiple objects (such as units in an RTS), the control object may be the **Camera** that determines the client's view of the world (although in general, the client's camera object does not need to be the same as the control object).

The object controlled by the client is important for several reasons:

1. The control object receives Moves from the client **GameConnection** (such as keyboard or gamepad input).
2. The control object is used to 'scope' the client, that is, to determine what else in the world is significant for the client, and which objects the server should give priority when sending network updates to the client.
Energy/Damage

ShapeBase includes basic energy and damage systems that may be used by derived classes as required. For example, the Player class uses energy to determine whether the character is capable of running and jumping, which can be used to mimic the character getting tired and having to rest before continuing. The Player class also uses the damage system PlayerData::onDestroyed callback to trigger a death animation. The Vehicle classes use the current damage level to trigger particle emitters, so a vehicle could progressively generate more smoke as it becomes more damaged.

ShapeBase also includes parameters to 'blow up' the object when it is Destroyed (damage level above ShapeBaseData::destroyedLevel). Blowing up an object can generate an explosion and debris, as well as exclude the object from rendering.

Parameters to control the object's energy and damage functionality can be found in the ShapeBaseData datablock.

See also:

ShapeBaseData::maxEnergy
ShapeBaseData::maxDamage
### Member Function Documentation

```cpp
void ShapeBase::applyDamage(float amount)
```

Increment the current damage level by the specified amount.

**Parameters:**

- `amount` value to add to current damage level

```cpp
bool ShapeBase::applyImpulse(Point3F pos, Point3F vec)
```

Apply an impulse to the object.

**Parameters:**

- `pos` world position of the impulse
- `vec` impulse momentum (velocity * mass)

**Returns:**

- `true`

```cpp
void ShapeBase::applyRepair(float amount)
```

Repair damage by the specified amount.

Note that the damage level is only reduced by repairRate per tick, so it may take several ticks for the total repair to complete.

**Parameters:**

- `amount` total repair value (subtracted from damage level over time)
bool ShapeBase::canCloak()

Check if this object can cloak.

**Returns:**
true

**Note:**
Not implemented as it always returns true.

void ShapeBase::changeMaterial(string mapTo, Material oldMat, Material newMat)

Change one of the materials on the shape.

This method changes materials per `mapTo` with others. The material that is being replaced is mapped to `unmapped_mat` as a part of this transition.

**Note:**
Warning, right now this only sort of works. It doesn't do a live update like it should.

**Parameters:**

- `mapTo` the name of the material target to remap (from `getTargetName`)
- `oldMat` the old `Material` that was mapped
- `newMat` the new `Material` to map

**Example:**

```cpp
// remap the first material in the shape
%mapTo = %obj.getTargetName( 0 );
%obj.changeMaterial( %mapTo, 0, MyMaterial );
```
bool ShapeBase::destroyThread(int slot)

Destroy an animation thread, which prevents it from playing.

**Parameters:**

  * slot  thread slot to destroy

**Returns:**

  * true if successful, false if failed

**See also:**

  * playThread

void ShapeBase::dumpMeshVisibility()

Print a list of visible and hidden meshes in the shape to the console for debugging purposes.

**Note:**

  * Only in a SHIPPING build.

Point3F ShapeBase::getAIRepairPoint()

Get the position at which the AI should stand to repair things.

If the shape defines a node called "AIRepairNode", this method will return the current world position of that node, otherwise "0 0 0".

**Returns:**

  * the AI repair position
float ShapeBase::getCameraFov( )

Returns the vertical field of view in degrees for this object if used as a camera.

**Returns:**
- current FOV as defined in `ShapeBaseData::cameraDefaultFov`

int ShapeBase::getControllingClient( )

Get the client (if any) that controls this object.

The controlling client is the one that will send moves to us to act on.

**Returns:**
- the ID of the controlling `GameConnection`, or 0 if this object is not controlled by any client.

**See also:**
- `GameConnection`

int ShapeBase::getControllingObject( )

Get the object (if any) that controls this object.

**Returns:**
- the ID of the controlling `ShapeBase` object, or 0 if this object is not controlled by another object.

float ShapeBase::getDamageFlash( )
Get the damage flash level.

**Returns:**
flash level

**See also:**
setDamageFlash

```c
float ShapeBase::getDamageLevel()
```

Get the object's current damage level.

**Returns:**
damage level

**See also:**
setDamageLevel()

```c
float ShapeBase::getDamagePercent()
```

Get the object's current damage level as a percentage of maxDamage.

**Returns:**
damageLevel / datablock.maxDamage

**See also:**
setDamageLevel()

```c
string ShapeBase::getDamageState()
```

Get the object's damage state.
Returns:
the damage state; one of "Enabled", "Disabled", "Destroyed"

See also:
setDamageState()

float ShapeBase::getDefaultCameraFov ( )

Returns the default vertical field of view in degrees for this object if used as a camera.

Returns:
Default FOV

float ShapeBase::getEnergyLevel ( )

Get the object's current energy level.

Returns:
energy level

See also:
setEnergyLevel()

float ShapeBase::getEnergyPercent ( )

Get the object's current energy level as a percentage of maxEnergy.

Returns:
energyLevel / datablock.maxEnergy

See also:
setEnergyLevel()

**Point3F ShapeBase::getEyePoint()**

Get the position of the 'eye' for this object.

If the object model has a node called 'eye', this method will return that node's current world position, otherwise it will return the object's current world position.

**Returns:**
the eye position for this object

**See also:**
getEyeVector
gEyeTransform

**TransformF ShapeBase::getEyeTransform()**

Get the 'eye' transform for this object.

If the object model has a node called 'eye', this method will return that node's current transform, otherwise it will return the object's current transform.

**Returns:**
the eye transform for this object

**See also:**
getEyeVector
gEyePoint

**VectorF ShapeBase::getEyeVector()**
Get the forward direction of the 'eye' for this object.

If the object model has a node called 'eye', this method will return that node's current forward direction vector, otherwise it will return the object's current forward direction vector.

**Returns:**
the eye vector for this object

**See also:**
getEyePoint
getEyeTransform

```cpp
bool ShapeBase::getImageAltTrigger(int slot)
```

Get the alt trigger state of the Image mounted in the specified slot.

**Parameters:**

`slot` Image slot to query

**Returns:**
the Image's current alt trigger state

```cpp
bool ShapeBase::getImageAmmo(int slot)
```

Get the ammo state of the Image mounted in the specified slot.

**Parameters:**

`slot` Image slot to query

**Returns:**
the Image's current ammo state
bool ShapeBase::getImageGenericTrigger ( int slot, int trigger )

Get the generic trigger state of the Image mounted in the specified slot.

**Parameters:**

- `slot` Image slot to query
- `trigger` Generic trigger number

**Returns:**

the Image's current generic trigger state

bool ShapeBase::getImageLoaded ( int slot )

Get the loaded state of the Image mounted in the specified slot.

**Parameters:**

- `slot` Image slot to query

**Returns:**

the Image's current loaded state

string ShapeBase::getImageScriptAnimPrefix ( int slot )

Get the script animation prefix of the Image mounted in the specified slot.

**Parameters:**

- `slot` Image slot to query

**Returns:**

the Image's current script animation prefix
int ShapeBase::getImageSkinTag(int slot )

Get the skin tag ID for the Image mounted in the specified slot.

Parameters:
   slot  Image slot to query

Returns:
   the skinTag value passed to mountImage when the image was mounted

string ShapeBase::getImageState(int slot )

Get the name of the current state of the Image in the specified slot.

Parameters:
   slot  Image slot to query

Returns:
   name of the current Image state, or "Error" if slot is invalid

bool ShapeBase::getImageTarget(int slot )

Get the target state of the Image mounted in the specified slot.

Parameters:
   slot  Image slot to query

Returns:
   the Image's current target state
bool ShapeBase::getImageTrigger(int slot )

Get the trigger state of the Image mounted in the specified slot.

**Parameters:**

`slot` Image slot to query

**Returns:**

the Image's current trigger state

string ShapeBase::getLookAtPoint(float distance = 2000,
                                int     typeMask = 0xFFFFFFFF )

Get the world position this object is looking at.

Casts a ray from the eye and returns information about what the ray hits.

**Parameters:**

`distance` maximum distance of the raycast
`typeMask` typeMask of objects to include for raycast collision testing

**Returns:**

look-at information as "Object HitX HitY HitZ [Material]" or empty string for no hit

**Example:**

```%
lookat = %obj.getLookAtPoint();
echo( "Looking at: " @ getWords( %lookat,
```

float ShapeBase::getMaxDamage( )
Get the object's maxDamage level.

**Returns:**

datablock.maxDamage

---

```cpp
string ShapeBase::getModelFile()
```

Get the model filename used by this shape.

**Returns:**

the shape filename

---

```cpp
int ShapeBase::getMountedImage(int slot)
```

Get the Image mounted in the specified slot.

**Parameters:**

- `slot` Image slot to query

**Returns:**

ID of the `ShapeBaseImageData` datablock mounted in the slot, or 0 if no Image is mounted there.

---

```cpp
int ShapeBase::getMountSlot(ShapeBaseImageData image)
```

Get the first slot the given datablock is mounted to on this object.

**Parameters:**

- `image` `ShapeBaseImageData` datablock to query

**Returns:**

index of the first slot the Image is mounted in, or -1 if the Image is not mounted in any slot on this object.
Point3F ShapeBase::getMuzzlePoint(int slot)

Get the muzzle position of the Image mounted in the specified slot.

If the Image shape contains a node called 'muzzlePoint', then the
muzzle position is the position of that node in world space. If no
such node is specified, the slot's mount node is used instead.

**Parameters:**

- *slot* Image slot to query

**Returns:**

- the muzzle position, or "0 0 0" if the slot is invalid

VectorF ShapeBase::getMuzzleVector(int slot)

Get the muzzle vector of the Image mounted in the specified slot.

If the Image shape contains a node called 'muzzlePoint', then the
muzzle vector is the forward direction vector of that node's
transform in world space. If no such node is specified, the slot's
mount node is used instead.

If the correctMuzzleVector flag (correctMuzzleVectorTP in 3rd
person) is set in the Image, the muzzle vector is computed to point
at whatever object is right in front of the object's 'eye' node.

**Parameters:**

- *slot* Image slot to query

**Returns:**

- the muzzle vector, or "0 1 0" if the slot is invalid

int ShapeBase::getPendingImage(int slot)
Get the Image that will be mounted next in the specified slot.

Calling `mountImage` when an Image is already mounted does one of two things:

1. Mount the new Image immediately, the old Image is discarded and whatever state it was in is ignored.
2. If the current Image state does not allow Image changes, the new Image is marked as pending, and will not be mounted until the current state completes. eg. if the user changes weapons, you may wish to ensure that the current weapon firing state plays to completion first.

This command retrieves the ID of the pending Image (2nd case above).

**Parameters:**

\[ slot \text{ Image slot to query} \]

**Returns:**

ID of the pending `ShapeBaseImageData` datablock, or 0 if none.

```c
float ShapeBase::getRechargeRate() {}
```

Get the current recharge rate.

**Returns:**

the recharge rate (per tick)

**See also:**

`setRechargeRate()`

```c
float ShapeBase::getRepairRate() {}
```
Get the per-tick repair amount.

**Returns:**
the current value to be subtracted from damage level each tick

**See also:**
setRepairRate

```cpp
string ShapeBase::getShapeName()
```

Get the name of the shape.

**Note:**
This is the name of the shape object that is sent to the client, not the DTS or DAE model filename.

**Returns:**
the name of the shape

**See also:**
setShapeName()

```cpp
string ShapeBase::getSkinName()
```

Get the name of the skin applied to this shape.

**Returns:**
the name of the skin

**See also:**
skin
setSkinName()
**TransformF ShapeBase::getSlotTransform(int slot)**

Get the world transform of the specified mount slot.

**Parameters:**

- `slot` Image slot to query

**Returns:**

the mount transform

**int ShapeBase::getTargetCount()**

Get the number of materials in the shape.

**Returns:**

the number of materials in the shape.

**See also:**

- `getTargetName()`

**string ShapeBase::getTargetName(int index)**

Get the name of the indexed shape material.

**Parameters:**

- `index` index of the material to get (valid range is 0 - `getTargetCount()-1`).

**Returns:**

the name of the indexed material.

**See also:**

- `getTargetCount()`
VectorF ShapeBase::getVelocity()

Get the object’s current velocity.

**Returns:**
the current velocity

Reimplemented in Camera.

float ShapeBase::getWhiteOut()

Get the white-out level.

**Returns:**
white-out level

**See also:**
setWhiteOut

bool ShapeBase::hasImageState(int slot, string state)

Check if the given state exists on the mounted Image.

**Parameters:**
slot Image slot to query
state Image state to check for

**Returns:**
true if the Image has the requested state defined.

bool ShapeBase::isCloaked()
Check if this object is cloaked.

**Returns:**
true if cloaked, false if not

**See also:**
setCloaked()

---

`bool ShapeBase::isDestroyed()`

Check if the object is in the Destroyed damage state.

**Returns:**
true if damage state is "Destroyed", false if not

**See also:**
isDisabled()
isEnabled()

---

`bool ShapeBase::isDisabled()`

Check if the object is in the Disabled or Destroyed damage state.

**Returns:**
true if damage state is not "Enabled", false if it is

**See also:**
isDestroyed()
isEnabled()

---

`bool ShapeBase::isEnabled()`
Check if the object is in the Enabled damage state.

**Returns:**
true if damage state is "Enabled", false if not

**See also:**
- isDestroyed()
- isDisabled()

```cpp
bool ShapeBase::isHidden( )
```

Check if the object is hidden.

**Returns:**
true if the object is hidden, false if visible.

```cpp
bool ShapeBase::isImageFiring( int slot )
```

Check if the current Image state is firing.

**Parameters:**
- `slot`  Image slot to query

**Returns:**
true if the current Image state in this slot has the 'stateFire' flag set.

```cpp
bool ShapeBase::isImageMounted( ShapeBaseImageData image )
```

Check if the given datablock is mounted to any slot on this object.

**Parameters:**
Returns:
true if the Image is mounted to any slot, false otherwise.

Mount a new Image.

Parameters:

- **image** the Image to mount
- **slot** Image slot to mount into (valid range is 0 - 3)
- **loaded** initial loaded state for the Image
- **skinTag** tagged string to reskin the mounted Image

Returns:
true if successful, false if failed

Example:

```plaintext
%player.mountImage( PistolImage, 1 );
%player.mountImage( CrossbowImage, 0, false );
%player.mountImage( RocketLauncherImage, 0 );
```

See also:
unmountImage()
getMountedImage()
getPendingImage()
isImageMounted()
Pause an animation thread.

If restarted using playThread, the animation will resume from the paused position.

**Parameters:**

*slot* thread slot to stop

**Returns:**

true if successful, false if failed

**See also:**

playThread

```cpp
bool ShapeBase::playAudio(int slot,
                        SFXTrack track)
```

Attach a sound to this shape and start playing it.

**Parameters:**

*slot* Audio slot index for the sound (valid range is 0 - 3)
*track* SFXTrack to play

**Returns:**

true if the sound was attached successfully, false if failed

**See also:**

stopAudio()

```cpp
bool ShapeBase::playThread(int slot,
                           string name ="
                           )
```
Start a new animation thread, or restart one that has been paused or stopped.

**Parameters:**

- **slot** thread slot to play. Valid range is 0 - 3) name of the animation sequence to play in this slot. If not specified, the paused or stopped thread in this slot will be resumed.

**Returns:**

true if successful, false if failed

**Example:**

```cpp
%obj.playThread( 0, "ambient" ); // F
%obj.setThreadTimeScale( 0, 0.5 ); // F
%obj.pauseThread( 0 ); // F
%obj.playThread( 0 ); // F
%obj.playThread( 0, "spin" ); // F
```

**See also:**

- pauseThread()
- stopThread()
- setThreadDir()
- setThreadTimeScale()
- destroyThread()

---

```cpp
void ShapeBase::setAllMeshesHidden(bool hide )
```

Set the hidden state on all the shape meshes.

This allows you to hide all meshes in the shape, for example, and then only enable a few.

**Parameters:**

- **hide** new hidden state for all meshes
void ShapeBase::setCameraFov(float fov)

Set the vertical field of view in degrees for this object if used as a camera.

**Parameters:**

- `fov` new FOV value

void ShapeBase::setCloaked(bool cloak)

Set the cloaked state of this object.

When an object is cloaked it is not rendered.

**Parameters:**

- `cloak` true to cloak the object, false to uncloak

See also:

- `isCloaked()`

void ShapeBase::setDamageFlash(float level)

Set the damage flash level.

Damage flash may be used as a postfx effect to flash the screen when the client is damaged.

**Note:**

- Relies on the flash postFx.

**Parameters:**

- `level` flash level (0-1)
void ShapeBase::setDamageLevel(float level)

Set the object's current damage level.

Parameters:

   level new damage level

See also:

generateFlash()
generateLevel()
generatePercent()

bool ShapeBase::setDamageState(string state)

Set the object's damage state.

Parameters:

   state should be one of "Enabled", "Disabled", "Destroyed"

Returns:

   true if successful, false if failed

See also:

generateState()
Parameters:

vec damage direction vector

Example:

```cpp
%obj.setDamageVector( "0 0 1" );
```

```cpp
void ShapeBase::setEnergyLevel(float level)
```

Set this object's current energy level.

**Parameters:**

level new energy level

**See also:**

- `getEnergyLevel()`
- `getEnergyPercent()`

```cpp
void ShapeBase:: setHidden(bool show)
```

Add or remove this object from the scene.

When removed from the scene, the object will not be processed or rendered.

**Parameters:**

show False to hide the object, true to re-show it

Reimplemented from `SimObject`.

```cpp
bool ShapeBase::setImageAltTrigger(int slot, bool state)
```
Set the alt trigger state of the Image mounted in the specified slot.

**Parameters:**
- *slot* Image slot to modify
- *state* new alt trigger state for the Image

**Returns:**
- the Image's new alt trigger state

```cpp
bool ShapeBase::setImageAmmo(int slot, bool state)
```

Set the ammo state of the Image mounted in the specified slot.

**Parameters:**
- *slot* Image slot to modify
- *state* new ammo state for the Image

**Returns:**
- the Image's new ammo state

```cpp
int ShapeBase::setImageGenericTrigger(int slot, int trigger, bool state)
```

Set the generic trigger state of the Image mounted in the specified slot.

**Parameters:**
- *slot* Image slot to modify
- *trigger* Generic trigger number
Returns:
the Image's new generic trigger state or -1 if there was a problem.

bool ShapeBase::setImageLoaded(int slot, bool state)

Set the loaded state of the Image mounted in the specified slot.

Parameters:
slot Image slot to modify
state new loaded state for the Image

Returns:
the Image's new loaded state

void ShapeBase::setImageScriptAnimPrefix(int slot, string prefix)

Set the script animation prefix for the Image mounted in the specified slot.

This is used to further modify the prefix used when deciding which animation sequence to play while this image is mounted.

Parameters:
slot Image slot to modify
prefix The prefix applied to the image
bool ShapeBase::setImageTarget (int slot,
        bool state
    )

Set the target state of the Image mounted in the specified slot.

**Parameters:**

- *slot*  Image slot to modify
- *state* new target state for the Image

**Returns:**

the Image's new target state

bool ShapeBase::setImageTrigger (int slot,
        bool state
    )

Set the trigger state of the Image mounted in the specified slot.

**Parameters:**

- *slot*  Image slot to modify
- *state* new trigger state for the Image

**Returns:**

the Image's new trigger state

void ShapeBase::setInvincibleMode (float time,
        float speed
    )

Setup the invincible effect.

This effect is used for HUD feedback to the user that they are invincible.
**Note:**
Currently not implemented

**Parameters:**

- *time* duration in seconds for the invincible effect
- *speed* speed at which the invincible effect progresses

```cpp
void ShapeBase::setMeshHidden(string name, bool hide)
```

Set the hidden state on the named shape mesh.

**Parameters:**

- *name* name of the mesh to hide/show
- *hide* new hidden state for the mesh

```cpp
void ShapeBase::setRechargeRate(float rate)
```

Set the recharge rate.

The recharge rate is added to the object's current energy level each tick, up to the maxEnergy level set in the `ShapeBaseData` datablock.

**Parameters:**

- *rate* the recharge rate (per tick)

**See also:**

- `getRechargeRate()`

```cpp
void ShapeBase::setRepairRate(float rate)
```
Set amount to repair damage by each tick.

Note that this value is separate to the repairRate field in ShapeBaseData. This value will be subtracted from the damage level each tick, whereas the ShapeBaseData field limits how much of the applyRepair value is subtracted each tick. Both repair types can be active at the same time.

**Parameters:**

\[
\text{rate} \quad \text{value to subtract from damage level each tick (must be } > 0)
\]

**See also:**

getRepairRate()

---

### void ShapeBase::setShapeName (string name )

Set the name of this shape.

**Note:**

This is the name of the shape object that is sent to the client, not the DTS or DAE model filename.

**Parameters:**

\[
\text{name} \quad \text{new name for the shape}
\]

**See also:**

getShapeName()

---

### void ShapeBase::setSkinName (string name )

Apply a new skin to this shape.

'Skinning' the shape effectively renames the material targets,
allowing different materials to be used on different instances of the same model.

**Parameters:**

\[ name \] name of the skin to apply

**See also:**

skin
getSkinName()

```cpp
bool ShapeBase::setThreadDir (int slot,
    bool fwd
)
```

Set the playback direction of an animation thread.

**Parameters:**

\[ slot \] thread slot to modify
\[ fwd \] true to play the animation forwards, false to play backwards

**Returns:**

true if successful, false if failed

**See also:**

playThread()

```cpp
bool ShapeBase::setThreadPosition (int slot,
    float pos
)
```

Set the position within an animation thread.

**Parameters:**
slot  thread slot to modify
pos  position within thread

Returns:
true if successful, false if failed

See also:
playThread

```cpp
bool ShapeBase::setThreadTimeScale (int slot, float scale)
```

Set the playback time scale of an animation thread.

Parameters:
- slot  thread slot to modify
- scale  new thread time scale (1=normal speed, 0.5=half speed etc)

Returns:
true if successful, false if failed

See also:
playThread

```cpp
bool ShapeBase::setVelocity (Point3F vel)
```

Set the object's velocity.

Parameters:
- vel  new velocity for the object

Returns:
void ShapeBase::setWhiteOut(float level)

Set the white-out level.

White-out may be used as a postfx effect to brighten the screen in response to a game event.

**Note:**

Relies on the flash postFx.

**Parameters:**

- `level` flash level (0-1)

**See also:**

- getWhiteOut()

void ShapeBase::startFade(int time,
                          int delay,
                          bool fadeOut)

Fade the object in or out without removing it from the scene.

A faded out object is still in the scene and can still be collided with, so if you want to disable collisions for this shape after it fades out use setHidden to temporarily remove this shape from the scene.

**Note:**

Items have the ability to light their surroundings. When an Item with an active light is fading out, the light it emits is correspondingly reduced until it goes out. Likewise, when the item fades in, the light is turned-up till it reaches its normal brightness.
Parameters:

- **time** duration of the fade effect in ms
- **delay** delay in ms before the fade effect begins
- **fadeOut** true to fade-out to invisible, false to fade-in to full visibility

```cpp
bool ShapeBase::stopAudio(int slot)
```

Stop a sound started with playAudio.

**Parameters:**

- **slot** audio slot index (started with playAudio)

**Returns:**

true if the sound was stopped successfully, false if failed

**See also:**

- `playAudio()`

```cpp
bool ShapeBase::stopThread(int slot)
```

Stop an animation thread.

If restarted using playThread, the animation will start from the beginning again.

**Parameters:**

- **slot** thread slot to stop

**Returns:**

true if successful, false if failed

**See also:**

- `playThread`
bool ShapeBase::unmountImage(int slot)

Unmount the mounted Image in the specified slot.

**Parameters:**

  * slot Image slot to unmount

**Returns:**

  * true if successful, false if failed

**See also:**

  * mountImage()

float ShapeBase::validateCameraFov(float fov)

Called on the server when the client has requested a FOV change.

When the client requests that its field of view should be changed (because they want to use a sniper scope, for example) this new FOV needs to be validated by the server. This method is called if it exists (it is optional) to validate the requested FOV, and modify it if necessary. This could be as simple as checking that the FOV falls within a correct range, to making sure that the FOV matches the capabilities of the current weapon.

Following this method, ShapeBase ensures that the given FOV still falls within the datablock's cameraMinFov and cameraMaxFov. If that is good enough for your purposes, then you do not need to define the validateCameraFov() callback for your ShapeBase.

**Parameters:**

  * fov The FOV that has been requested by the client.

**Returns:**

  * The FOV as validated by the server.
See also:

ShapeBaseData
Member Data Documentation

bool ShapeBase::isAIControlled

Is this object AI controlled.

If True then this object is considered AI controlled and not player controlled.

string ShapeBase::skin

The skin applied to the shape.

'Skinning' the shape effectively renames the material targets, allowing different materials to be used on different instances of the same model. Using getSkinName() and setSkinName() is equivalent to reading and writing the skin field directly.

Any material targets that start with the old skin name have that part of the name replaced with the new skin name. The initial old skin name is "base". For example, if a new skin of "blue" was applied to a model that had material targets base_body and face, the new targets would be blue_body and face. Note that face was not renamed since it did not start with the old skin name of "base".

To support models that do not use the default "base" naming convention, you can also specify the part of the name to replace in the skin field itself. For example, if a model had a material target called shapemat, we could apply a new skin "shape=blue", and the material target would be renamed to bluemat (note "shape" has been replaced with "blue").

Multiple skin updates can also be applied at the same time by separating them with a semicolon. For example: "base=blue;face=happy_face".

Material targets are only renamed if an existing Material maps to
that name, or if there is a diffuse texture in the model folder with the same name as the new target.
ShapeBaseData Class Reference
[Game Objects]

Defines properties for a ShapeBase object. More...

Inheritance diagram for ShapeBaseData:

List of all members.
Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool checkDeployPos (TransformF txfm)</td>
<td>Check if there is the space at the given transform is free to spawn into.</td>
</tr>
</tbody>
</table>

TransformF getDeployTransform (Point3F pos, Point3F normal)  
Helper method to get a transform from a position and vector (suitable for use with setTransform).

Callbacks

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void onCollision (ShapeBase obj, SceneObject collObj, VectorF vec, float len)</td>
<td>Called when we collide with another object.</td>
</tr>
<tr>
<td>void onDamage (ShapeBase obj, float delta)</td>
<td>Called when the object is damaged.</td>
</tr>
<tr>
<td>void onDestroyed (ShapeBase obj, string lastState)</td>
<td>Called when the object damage state changes to Destroyed.</td>
</tr>
<tr>
<td>void onDisabled (ShapeBase obj, string lastState)</td>
<td>Called when the object damage state changes to Disabled.</td>
</tr>
<tr>
<td>void onEnabled (ShapeBase obj, string lastState)</td>
<td>Called when the object damage state changes to Enabled.</td>
</tr>
<tr>
<td>void onEndSequence (ShapeBase obj, int slot)</td>
<td>Called when a thread playing a non-cyclic sequence reaches the end of the sequence.</td>
</tr>
<tr>
<td>void onForceUncloak (ShapeBase obj, string reason)</td>
<td>Called when the object is forced to uncloak.</td>
</tr>
<tr>
<td>void onImpact (ShapeBase obj, SceneObject collObj, VectorF vec, float len)</td>
<td>Called when we collide with another object beyond</td>
</tr>
</tbody>
</table>
some impact speed.

**void onTrigger (ShapeBase obj, int index, bool state)**
Called when a move trigger input changes state.
Public Attributes

Camera

*The settings used by the shape when it is the camera.*

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>cameraDefaultFov</td>
<td>The default camera vertical FOV in degrees.</td>
</tr>
<tr>
<td>float</td>
<td>cameraMaxDist</td>
<td>The maximum distance from the camera to the object.</td>
</tr>
<tr>
<td>float</td>
<td>cameraMaxFov</td>
<td>The maximum camera vertical FOV allowed in degrees.</td>
</tr>
<tr>
<td>float</td>
<td>cameraMinDist</td>
<td>The minimum distance from the camera to the object.</td>
</tr>
<tr>
<td>float</td>
<td>cameraMinFov</td>
<td>The minimum camera vertical FOV allowed in degrees.</td>
</tr>
<tr>
<td>bool</td>
<td>firstPersonOnly</td>
<td>Flag controlling whether the view from this object is first person only.</td>
</tr>
<tr>
<td>bool</td>
<td>observeThroughObject</td>
<td>Observe this object through its camera transform and default fov.</td>
</tr>
<tr>
<td>bool</td>
<td>useEyePoint</td>
<td>Flag controlling whether the client uses this object's eye point to view from.</td>
</tr>
</tbody>
</table>

Misc

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>computeCRC</td>
<td></td>
</tr>
</tbody>
</table>

If true, verify that the CRC of the client's shape model matches the server's CRC for the shape model when loaded by the client.

Reflection

<table>
<thead>
<tr>
<th>string</th>
<th>cubeReflectorDesc</th>
</tr>
</thead>
<tbody>
<tr>
<td>References a <code>ReflectorDesc</code> datablock that defines performance and quality properties for dynamic reflections.</td>
<td></td>
</tr>
</tbody>
</table>

Destruction

*Parameters related to the destruction effects of this object.*

<table>
<thead>
<tr>
<th>DebrisData</th>
<th>Debris</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debris to generate when this shape is blown up.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>filename</th>
<th>debrisShapeName</th>
</tr>
</thead>
<tbody>
<tr>
<td>The DTS or DAE model to use for auto-generated breakups.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ExplosionData</th>
<th>Explosion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explosion to generate when this shape is blown up.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>renderWhenDestroyed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Whether to render the shape when it is in the &quot;Destroyed&quot; damage state.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ExplosionData</th>
<th>underwaterExplosion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explosion to generate when this shape is blown up underwater.</td>
<td></td>
</tr>
</tbody>
</table>

Physics

<table>
<thead>
<tr>
<th>float</th>
<th>density</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shape density.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>drag</th>
</tr>
</thead>
</table>
### Drag factor.

<table>
<thead>
<tr>
<th>float</th>
<th>mass</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Shape mass.</td>
</tr>
</tbody>
</table>

### Damage/Energy

<table>
<thead>
<tr>
<th>float</th>
<th>destroyedLevel</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Damage level above which the object is destroyed.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>disabledLevel</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Damage level above which the object is disabled.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>inheritEnergyFromMount</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Flag controlling whether to manage our own energy level, or to use the energy level of the object we are mounted to.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>isInvincible</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Invincible flag; when invincible, the object cannot be damaged or repaired.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>maxDamage</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maximum damage level for this object.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>maxEnergy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maximum energy level for this object.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>repairRate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Rate at which damage is repaired in damage units/tick.</td>
</tr>
</tbody>
</table>

### Shadows

<table>
<thead>
<tr>
<th>bool</th>
<th>shadowEnable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Enable shadows for this shape (currently unused, shadows are always enabled).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>shadowMaxVisibleDistance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maximum distance at which shadow is visible (currently unused).</td>
</tr>
<tr>
<td>Type</td>
<td>Variable</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>float</td>
<td><code>shadowProjectionDistance</code></td>
</tr>
<tr>
<td>int</td>
<td><code>shadowSize</code></td>
</tr>
<tr>
<td>float</td>
<td><code>shadowSphereAdjust</code></td>
</tr>
</tbody>
</table>

**Render**

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filename</td>
<td><code>shapeFile</code></td>
<td>The DTS or DAE model to use for this object.</td>
</tr>
</tbody>
</table>
Detailed Description

Defines properties for a ShapeBase object.

See also:
  ShapeBase
Member Function Documentation

```cpp
bool ShapeBaseData::checkDeployPos(TransformF txfm)
```

Check if there is the space at the given transform is free to spawn into.

The shape's bounding box volume is used to check for collisions at the given world transform. Only interior and static objects are checked for collision.

**Parameters:**

- `txfm` Deploy transform to check

**Returns:**

True if the space is free, false if there is already something in the way.

**Note:**

This is a server side only check, and is not actually limited to spawning.

```cpp
TransformF ShapeBaseData::getDeployTransform(Point3F pos, Point3F normal)
```

Helper method to get a transform from a position and vector (suitable for use with setTransform).

**Parameters:**

- `pos` Desired transform position
- `normal` Vector of desired direction

**Returns:**

The deploy transform
void ShapeBaseData::onCollision (ShapeBase obj, SceneObject collObj, VectorF vec, float len)

Called when we collide with another object.

**Parameters:**

- **obj** The `ShapeBase` object
- **collObj** The object we collided with
- **vec** Collision impact vector
- **len** Length of the impact vector

void ShapeBaseData::onDamage (ShapeBase obj, float delta)

Called when the object is damaged.

**Parameters:**

- **obj** The `ShapeBase` object
- **obj** The `ShapeBase` object
- **delta** The amount of damage received

void ShapeBaseData::onDestroyed (ShapeBase obj, string lastState)

Called when the object damage state changes to Destroyed.

**Parameters:**
The `ShapeBase` object

`lastState` The previous damage state

```cpp
void ShapeBaseData::onDisabled(
    ShapeBase obj,
    string lastState
)
```

Called when the object damage state changes to Disabled.

**Parameters:**

- `obj` The `ShapeBase` object
- `lastState` The previous damage state

```cpp
void ShapeBaseData::onEnabled(
    ShapeBase obj,
    string lastState
)
```

Called when the object damage state changes to Enabled.

**Parameters:**

- `obj` The `ShapeBase` object
- `lastState` The previous damage state

```cpp
void ShapeBaseData::onEndSequence(
    ShapeBase obj,
    int slot
)
```

Called when a thread playing a non-cyclic sequence reaches the end of the sequence.

**Parameters:**

- `obj` The `ShapeBase` object
- `slot` Thread slot that finished playing
void ShapeBaseData::onForceUncloak(ShapeBase obj,  
  string reason)  

Called when the object is forced to uncloak.

**Parameters:**

- *obj* The `ShapeBase` object
- *reason* String describing why the object was uncloaked

void ShapeBaseData::onImpact(ShapeBase obj,  
  SceneObject collObj,  
  VectorF vec,  
  float len)  

Called when we collide with another object beyond some impact speed.

The `Player` class makes use of this callback when a collision speed is more than `PlayerData::minImpactSpeed`.

**Parameters:**

- *obj* The `ShapeBase` object
- *collObj* The object we collided with
- *vec* Collision impact vector
- *len* Length of the impact vector

void ShapeBaseData::onTrigger(ShapeBase obj,  
  int index,  
  bool state)  

Called when a move trigger input changes state.

**Parameters:**

- **obj**  The ShapeBase object
- **index**  Index of the trigger that changed
- **state**  New state of the trigger
**Member Data Documentation**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>ShapeBaseData::cameraDefaultFov</code></td>
</tr>
<tr>
<td></td>
<td>The default camera vertical FOV in degrees.</td>
</tr>
<tr>
<td>float</td>
<td><code>ShapeBaseData::cameraMaxDist</code></td>
</tr>
<tr>
<td></td>
<td>The maximum distance from the camera to the object.</td>
</tr>
<tr>
<td></td>
<td>Used when computing a custom camera transform for this object.</td>
</tr>
<tr>
<td></td>
<td>See also: <code>observeThroughObject</code></td>
</tr>
<tr>
<td>float</td>
<td><code>ShapeBaseData::cameraMaxFov</code></td>
</tr>
<tr>
<td></td>
<td>The maximum camera vertical FOV allowed in degrees.</td>
</tr>
<tr>
<td>float</td>
<td><code>ShapeBaseData::cameraMinDist</code></td>
</tr>
<tr>
<td></td>
<td>The minimum distance from the camera to the object.</td>
</tr>
<tr>
<td></td>
<td>Used when computing a custom camera transform for this object.</td>
</tr>
<tr>
<td></td>
<td>See also: <code>observeThroughObject</code></td>
</tr>
<tr>
<td>float</td>
<td><code>ShapeBaseData::cameraMinFov</code></td>
</tr>
<tr>
<td></td>
<td>The minimum camera vertical FOV allowed in degrees.</td>
</tr>
<tr>
<td>bool ShapeBaseData::computeCRC</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td></td>
</tr>
<tr>
<td>If true, verify that the CRC of the client's shape model matches the server's CRC for the shape model when loaded by the client.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string ShapeBaseData::cubeReflectorDesc</th>
</tr>
</thead>
<tbody>
<tr>
<td>References a ReflectorDesc datablock that defines performance and quality properties for dynamic reflections.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DebrisData ShapeBaseData::Debris</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debris to generate when this shape is blown up.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>filename ShapeBaseData::debrisShapeName</th>
</tr>
</thead>
<tbody>
<tr>
<td>The DTS or DAE model to use for auto-generated breakups.</td>
</tr>
</tbody>
</table>

**Note:**

- may not be functional.

<table>
<thead>
<tr>
<th>float ShapeBaseData::density</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shape density.</td>
</tr>
<tr>
<td>Used when computing buoyancy when in water.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float ShapeBaseData::destroyedLevel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Damage level above which the object is destroyed.</td>
</tr>
</tbody>
</table>
When the damage level increases above this value, the object damage state is set to "Destroyed".

<table>
<thead>
<tr>
<th>float ShapeBaseData::disabledLevel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Damage level above which the object is disabled. Currently unused.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float ShapeBaseData::drag</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drag factor. Reduces velocity of moving objects.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ExplosionData ShapeBaseData::Explosion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explosion to generate when this shape is blown up.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool ShapeBaseData::firstPersonOnly</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flag controlling whether the view from this object is first person only.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool ShapeBaseData::inheritEnergyFromMount</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flag controlling whether to manage our own energy level, or to use the energy level of the object we are mounted to.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool ShapeBaseData::isInvincible</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>
Invincible flag; when invincible, the object cannot be damaged or repaired.

float ShapeBaseData::mass

Shape mass. Used in simulation of moving objects.

float ShapeBaseData::maxDamage

Maximum damage level for this object.

float ShapeBaseData::maxEnergy

Maximum energy level for this object.

bool ShapeBaseData::observeThroughObject

Observe this object through its camera transform and default fov. If true, when this object is the camera it can provide a custom camera transform and FOV (instead of the default eye transform).

bool ShapeBaseData::renderWhenDestroyed

Whether to render the shape when it is in the "Destroyed" damage state.

float ShapeBaseData::repairRate
Rate at which damage is repaired in damage units/tick.
This value is subtracted from the damage level until it reaches 0.

`bool ShapeBaseData::shadowEnable`
Enable shadows for this shape (currently unused, shadows are always enabled).

`float ShapeBaseData::shadowMaxVisibleDistance`
Maximum distance at which shadow is visible (currently unused).

`float ShapeBaseData::shadowProjectionDistance`
Maximum height above ground to project shadow. If the object is higher than this no shadow will be rendered.

`int ShapeBaseData::shadowSize`
Size of the projected shadow texture (must be power of 2).

`float ShapeBaseData::shadowSphereAdjust`
Scalar applied to the radius of spot shadows (initial radius is based on the shape bounds but can be adjusted with this field).

`filename ShapeBaseData::shapeFile`
The DTS or DAE model to use for this object.
ExplosionData ShapeBaseData::underwaterExplosion

Explosion to generate when this shape is blown up underwater.

bool ShapeBaseData::useEyePoint

Flag controlling whether the client uses this object's eye point to view from.
ShapeBaseImageData Class Reference
[Game Objects]

Represents geometry to be mounted to a ShapeBase object. More...

Inheritance diagram for ShapeBaseImageData:

List of all members.
Public Member Functions

Callbacks

<table>
<thead>
<tr>
<th>void onMount (ShapeBase obj, int slot, float dt)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Called when the Image is first mounted to the object.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void onUnmount (ShapeBase obj, int slot, float dt)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Called when the Image is unmounted from the object.</td>
</tr>
</tbody>
</table>
### Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>accuFire</td>
<td>Flag to control whether the Image’s aim is automatically converged with the crosshair.</td>
</tr>
<tr>
<td>bool</td>
<td>animateAllShapes</td>
<td>Indicates that all shapes should be animated in sync.</td>
</tr>
<tr>
<td>bool</td>
<td>animateOnServer</td>
<td>Indicates that the image should be animated on the server.</td>
</tr>
<tr>
<td>Point3F</td>
<td>camShakeAmp</td>
<td>Amplitude of the camera shaking effect.</td>
</tr>
<tr>
<td>Point3F</td>
<td>camShakeFreq</td>
<td>Frequency of the camera shaking effect.</td>
</tr>
<tr>
<td>DebrisData</td>
<td>casing</td>
<td>DebrisData datablock to use for ejected casings.</td>
</tr>
<tr>
<td>bool</td>
<td>cloakable</td>
<td>Whether this Image can be cloaked.</td>
</tr>
<tr>
<td>bool</td>
<td>computeCRC</td>
<td>If true, verify that the CRC of the client's Image matches the server's CRC for the Image when loaded by the client.</td>
</tr>
<tr>
<td>bool</td>
<td>correctMuzzleVector</td>
<td>Flag to adjust the aiming vector to the eye's LOS point.</td>
</tr>
<tr>
<td>bool</td>
<td>correctMuzzleVectorTP</td>
<td></td>
</tr>
</tbody>
</table>
Flag to adjust the aiming vector to the camera's LOS point when in 3rd person view.

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>eMap</td>
<td>Whether to enable environment mapping on this Image.</td>
</tr>
<tr>
<td>MatrixPosition</td>
<td>eyeOffset</td>
<td>&quot;X Y Z&quot; translation offset from the ShapeBase model's eye node.</td>
</tr>
<tr>
<td>MatrixRotation</td>
<td>eyeRotation</td>
<td>&quot;X Y Z ANGLE&quot; rotation offset from the ShapeBase model's eye node.</td>
</tr>
<tr>
<td>bool</td>
<td>firstPerson</td>
<td>Set to true to render the image in first person.</td>
</tr>
<tr>
<td>caseString</td>
<td>imageAnimPrefix</td>
<td>Passed along to the mounting shape to modify animation sequences played in third person. [optional].</td>
</tr>
<tr>
<td>caseString</td>
<td>imageAnimPrefixFP</td>
<td>Passed along to the mounting shape to modify animation sequences played in first person. [optional].</td>
</tr>
<tr>
<td>float</td>
<td>lightBrightness</td>
<td>Brightness of the light this Image emits.</td>
</tr>
<tr>
<td>ColorF</td>
<td>lightColor</td>
<td>The color of light this Image emits.</td>
</tr>
<tr>
<td>int</td>
<td>lightDuration</td>
<td>Duration in SimTime of Pulsing and WeaponFire type lights.</td>
</tr>
<tr>
<td>Type</td>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>--------------------</td>
<td>---------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>float</td>
<td>lightRadius</td>
<td>Radius of the light this Image emits.</td>
</tr>
<tr>
<td>ShapeBaseImageLightType</td>
<td>lightType</td>
<td>The type of light this Image emits.</td>
</tr>
<tr>
<td>float</td>
<td>mass</td>
<td>Mass of this Image.</td>
</tr>
<tr>
<td>int</td>
<td>maxConcurrentSounds</td>
<td>Maximum number of sounds this Image can play at a time.</td>
</tr>
<tr>
<td>float</td>
<td>minEnergy</td>
<td>Minimum Image energy for it to be operable.</td>
</tr>
<tr>
<td>int</td>
<td>mountPoint</td>
<td>Mount node # to mount this Image to.</td>
</tr>
<tr>
<td>MatrixPosition</td>
<td>offset</td>
<td>&quot;X Y Z&quot; translation offset from this Image's mountPoint node to attach to.</td>
</tr>
<tr>
<td>ProjectileData</td>
<td>Projectile</td>
<td>The projectile fired by this Image.</td>
</tr>
<tr>
<td>MatrixRotation</td>
<td>rotation</td>
<td>&quot;X Y Z ANGLE&quot; rotation offset from this Image's mountPoint node to attach to.</td>
</tr>
<tr>
<td>float</td>
<td>scriptAnimTransitionTime</td>
<td>The amount of time to transition between the previous sequence and new sequence when the script prefix has changed.</td>
</tr>
<tr>
<td>bool</td>
<td>shakeCamera</td>
<td></td>
</tr>
<tr>
<td><strong>Flag indicating whether the camera should shake when this Image fires.</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>filename</strong></td>
<td><strong>shapeFile</strong></td>
<td></td>
</tr>
<tr>
<td>The DTS or DAE model to use for this Image.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| **filename** | **shapeFileFP** |
|---------------------------------------------------------------|
| The DTS or DAE model to use for this Image when in first person. |

| **Point3F** | **shellExitDir** |
|---------------------------------------------------------------|
| Vector direction to eject shell casings. |

| **float** | **shellExitVariance** |
|---------------------------------------------------------------|
| Variance (in degrees) from the shellExitDir vector to eject casings. |

| **float** | **shellVelocity** |
|---------------------------------------------------------------|
| Speed at which to eject casings. |

| **bool** | **stateAllowImageChange [31]** |
|---------------------------------------------------------------|
| If false, other Images will temporarily be blocked from mounting while the state machine is executing the tasks in this state. |

| **bool** | **stateAlternateFire [31]** |
|---------------------------------------------------------------|
| The first state with this set to true is the state entered by the client when it receives the 'altFire' event. |

| **bool** | **stateDirection [31]** |
|---------------------------------------------------------------|
| Direction of the animation to play in this state. |

<p>| <strong>bool</strong> | <strong>stateEjectShell [31]</strong> |
|---------------------------------------------------------------|
| If true, a shell casing will be ejected in this state. |</p>
<table>
<thead>
<tr>
<th><strong>ParticleEmitterData</strong></th>
<th><strong>stateEmitter</strong> [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Emitter to generate particles in this state (from muzzle point or specified node).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>string</strong></th>
<th><strong>stateEmitterNode</strong> [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Name of the node to emit particles from.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong></th>
<th><strong>stateEmitterTime</strong> [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>How long (in seconds) to emit particles on entry to this state.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float</strong></th>
<th><strong>stateEnergyDrain</strong> [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Amount of energy to subtract from the Image in this state.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>bool</strong></th>
<th><strong>stateFire</strong> [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The first state with this set to true is the state entered by the client when it receives the 'fire' event.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>bool</strong></th>
<th><strong>stateIgnoreLoadedForReady</strong> [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If set to true, and both ready and loaded transitions are true, the ready transition will be taken instead of the loaded transition.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>ShapeBaseImageLoadedState</strong></th>
<th><strong>stateLoadedFlag</strong> [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Set the loaded state of the Image.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>caseString</strong></th>
<th><strong>stateName</strong> [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Name of this state.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>ShapeBaseImageRecoilState</strong></th>
<th><strong>stateRecoil</strong> [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Type of recoil sequence to play on the ShapeBase object on entry to this state.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>bool</strong></th>
<th><strong>stateReload</strong> [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The first state with this set to true is the state entered by the client.</td>
</tr>
</tbody>
</table>
when it receives the 'reload' event.

<table>
<thead>
<tr>
<th>bool</th>
<th>stateScaleAnimation [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td>If true, the timeScale of the stateSequence animation will be adjusted such that the sequence plays for stateTimeoutValue seconds.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>stateScaleAnimationFP [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td>If true, the timeScale of the first person stateSequence animation will be adjusted such that the sequence plays for stateTimeoutValue seconds.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>stateScaleShapeSequence [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indicates if the sequence to be played on the mounting shape should be scaled to the length of the state.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>caseString</th>
<th>stateScript [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method to execute on entering this state.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>stateSequence [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of the sequence to play on entry to this state.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>stateSequenceNeverTransition [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Never allow a transition to this sequence. Often used for a fire sequence.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>stateSequenceRandomFlash [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td>If true, the muzzle flash sequence will be played while in this state.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>stateSequenceTransitionIn [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Do we transition to the state's sequence when we enter the</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Property</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>bool</td>
<td><code>stateSequenceTransitionOut [31]</code></td>
</tr>
<tr>
<td>float</td>
<td><code>stateSequenceTransitionTime [31]</code></td>
</tr>
<tr>
<td>string</td>
<td><code>stateShapeSequence [31]</code></td>
</tr>
<tr>
<td>SFXTrack</td>
<td><code>stateSound [31]</code></td>
</tr>
<tr>
<td>ShapeBaseImageSpinState</td>
<td><code>stateSpinThread [31]</code></td>
</tr>
<tr>
<td>float</td>
<td><code>stateTimeoutValue [31]</code></td>
</tr>
<tr>
<td>string</td>
<td><code>stateTransitionGeneric0In [31]</code></td>
</tr>
<tr>
<td>string</td>
<td><code>stateTransitionGeneric0Out [31]</code></td>
</tr>
<tr>
<td>string</td>
<td><code>stateTransitionGeneric1In [31]</code></td>
</tr>
<tr>
<td>String</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>stateTransitionGeneric1Out</code> [31]</td>
<td>Name of the state to transition to when the generic trigger 1 state changes to false.</td>
</tr>
<tr>
<td><code>stateTransitionGeneric2In</code> [31]</td>
<td>Name of the state to transition to when the generic trigger 2 state changes to true.</td>
</tr>
<tr>
<td><code>stateTransitionGeneric2Out</code> [31]</td>
<td>Name of the state to transition to when the generic trigger 2 state changes to false.</td>
</tr>
<tr>
<td><code>stateTransitionGeneric3In</code> [31]</td>
<td>Name of the state to transition to when the generic trigger 3 state changes to true.</td>
</tr>
<tr>
<td><code>stateTransitionGeneric3Out</code> [31]</td>
<td>Name of the state to transition to when the generic trigger 3 state changes to false.</td>
</tr>
<tr>
<td><code>stateTransitionOnAltTriggerDown</code> [31]</td>
<td>Name of the state to transition to when the alt trigger state of the Image changes to false (alt fire button up).</td>
</tr>
<tr>
<td><code>stateTransitionOnAltTriggerUp</code> [31]</td>
<td>Name of the state to transition to when the alt trigger state of the Image changes to true (alt fire button down).</td>
</tr>
<tr>
<td><code>stateTransitionOnAmmo</code> [31]</td>
<td>Name of the state to transition to when the ammo state of the Image...</td>
</tr>
<tr>
<td>string</td>
<td>stateTransitionOnLoaded [31]</td>
</tr>
<tr>
<td>--------</td>
<td>----------------------------</td>
</tr>
<tr>
<td></td>
<td>Name of the state to transition to when the loaded state of the Image changes to 'Loaded'.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>stateTransitionOnMotion [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Name of the state to transition to when the Player moves.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>stateTransitionOnNoAmmo [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Name of the state to transition to when the ammo state of the Image changes to false.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>stateTransitionOnNoMotion [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Name of the state to transition to when the Player stops moving.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>stateTransitionOnNoTarget [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Name of the state to transition to when the Image loses a target.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>stateTransitionOnNotLoaded [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Name of the state to transition to when the loaded state of the Image changes to 'Empty'.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>stateTransitionOnNotWet [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Name of the state to transition to when the Image exits the water.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>stateTransitionOnTarget [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Name of the state to transition to when the Image gains a target.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>stateTransitionOnTimeout [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Name of the state to transition to when we have been in this state for stateTimeoutValue seconds.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>stateTransitionOnTriggerDown [31]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Changes to true.</td>
</tr>
<tr>
<td><strong>Name of the state to transition to when the trigger state of the Image changes to false (fire button released).</strong></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td></td>
</tr>
<tr>
<td><strong>string</strong> stateTransitionOnTriggerUp [31]</td>
<td></td>
</tr>
<tr>
<td><strong>String</strong> stateTransitionOnWet [31] Name of the state to transition to when the trigger state of the Image changes to true (fire button down).</td>
<td></td>
</tr>
<tr>
<td><strong>string</strong> stateTransitionOnWet [31]</td>
<td></td>
</tr>
<tr>
<td><strong>bool</strong> stateWaitForTimeout [31] If false, this state ignores stateTimeoutValue and transitions immediately if other transition conditions are met.</td>
<td></td>
</tr>
<tr>
<td><strong>bool</strong> useEyeNode Mount image using image's eyeMount node and place the camera at the image's eye node (or at the eyeMount node if the eye node is missing).</td>
<td></td>
</tr>
<tr>
<td><strong>bool</strong> useRemainderDT If true, allow multiple timeout transitions to occur within a single tick (useful if states have a very small timeout).</td>
<td></td>
</tr>
<tr>
<td><strong>bool</strong> usesEnergy Flag indicating whether this Image uses energy instead of ammo. The energy level comes from the ShapeBase object we're mounted to.</td>
<td></td>
</tr>
</tbody>
</table>
**Detailed Description**

Represents geometry to be mounted to a *ShapeBase* object.

Unlike other datablocks, *ShapeBaseImageData* does not have a base class associated with it. Instead, this datablock is an abstraction of geometry that can only be mounted to a *ShapeBase* object and is only used by a ShapeBase object.

The most common use for *ShapeBaseImageData* objects (referred to as Images hereafter) is for weapons carried by a *Player* or *Vehicle* object, and much of the functionality provided by the Image is aimed at that use-case. Images include a powerful state machine to control animations, sounds, script callbacks, and state transitions. This state system is downloaded to the client so that clients can predict state changes and animate accordingly.

The following example - a grenade launcher weapon - demonstrates the flexibility of the system. The weapon includes states and transitions to handle the normal ready->fire->reload->ready loop as well as noammo->dryfire for firing when the weapon is out of ammo.
Example:

datablock ShapeBaseImageData(GrenadeLauncher) {

    // Basic properties
    shapefile = "art/shapes/weapons/ramrifle/base.dts"

    // Specify mount point & offset for 3rd
    // for first person rendering.
mountPoint = 0;
offset = "0.0 0.0 0.1";
eyeOffset = "0.25 0.4 -0.4";

// Add the WeaponImage namespace as a parent, provides some hooks into the inventory system.
className = "WeaponImage";

// Projectile and Ammo.
item = GrenadeLauncher;
ammo = GrenadeLauncherAmmo;
projectile = GrenadeLauncherProjectile;
wetProjectile = GrenadeWetProjectile;
projectileType = Projectile;

// Shell casings
casing = GrenadeLauncherShellCasing;
shellExitDir = "1.0 0.3 1.0";
shellExitOffset = "0.15 -0.56 -0.1";
shellExitVariance = 15.0;
shellVelocity = 3.0;

// Let there be light - NoLight, ConstantLight, PulsingLight, WeaponFireLight.
lightType = "WeaponFireLight";
lightColor = "1.0 1.0 0.9";
lightDuration = 200;
lightRadius = 20;

// Initial start up state
stateName[0] = "Preactivate";
stateTransitionOnLoaded[0] = "Activate"
stateTransitionOnNoAmmo[0] = "NoAmmo";
// Activating the gun.
// Called when the weapon is first mounted
stateName[1] = "Activate";
stateTransitionOnTimeout[1] = "Ready";
stateTimeoutValue[1] = 0.6;
stateSequence[1] = "Activate";

// Ready to fire, just waiting for the trigger
stateName[2] = "Ready";
stateTransitionOnNoAmmo[2] = "NoAmmo";
stateTransitionOnTriggerDown[2] = "CheckWet";
stateSequence[2] = "Ready";

// Fire the weapon. Calls the fire script
stateName[3] = "Fire";
stateTransitionOnTimeout[3] = "PostFire";
stateTimeoutValue[3] = 0.4;
stateFire[3] = true;
stateAllowImageChange[3] = false;
stateSequence[3] = "Fire";
stateScript[3] = "onFire";

// Check ammo
stateName[4] = "PostFire";
stateTransitionOnAmmo[4] = "Reload";
stateTransitionOnNoAmmo[4] = "NoAmmo";

// Play the reload animation, and transition
stateName[5] = "Reload";
stateTransitionOnTimeout[5] = "Ready";
stateTimeoutValue[5] = 0.2;
stateAllowImageChange[5] = false;
stateSequence[5] = "Reload";
stateEjectShell[5] = false; // set to true to enable shell casing eject

// No ammo in the weapon, just idle until something shows up.
// Play the dry fire sound if the trigger is pulled.
stateName[6] = "NoAmmo";
stateTransitionOnAmmo[6] = "Reload";
stateSequence[6] = "NoAmmo";

// No ammo dry fire
stateName[7] = "DryFire";
stateTimeoutValue[7] = 1.0;
stateTransitionOnTimeout[7] = "NoAmmo";

// Check if wet
stateName[8] = "CheckWet";
stateTransitionOnWet[8] = "WetFire";
stateTransitionOnNotWet[8] = "Fire";

// Wet fire
stateName[9] = "WetFire";
stateTransitionOnTimeout[9] = "PostFire";
stateTimeoutValue[9] = 0.4;
stateFire[9] = true;
stateAllowImageChange[9] = false;
stateSequence[9] = "Fire";
stateScript[9] = "onWetFire";
};
Images are mounted into a slot on the target `ShapeBase` derived object as shown below.

**Example:**

```cpp
$WeaponSlot = 0;

...

// Use a weapon by mounting it onto the given object.
// %data is the weapon whose .image member points to its ShapeBaseImageData datablock.
// %obj is the object to mount the weapon onto.

function Weapon::onUse( %data, %obj )
{
    // Default behavior for all weapons is to mount it into the object's weapon slot, as defined by $WeaponSlot here.
    // checking if the requested weapon is already mounted.
    if ( %obj.getMountedImage($WeaponSlot)
    {
        // The requested weapon is not mounted.
        %obj.mountImage( %data.image, $WeaponSlot
    }
}
```
Weapon Shape Nodes

The DTS or DAE model used for the Image has the following requirements:

**ejectPoint node**
- Node from which shell casings are emitted

**muzzlePoint node**
- Node used to fire projectiles and particles

**retractionPoint node**
- Nearest point to use as muzzle when up against a wall (and muzzle node is inside wall). This node is optional.

**mountPoint node**
- Where to attach to on this object. If not present the origin of the model will be used.

**stateEmitterNode**
- Each state may optionally have a particle emitter, which will play at this node's location. One example is smoke coming from the barrel of a gun after it has been fired. If this node is not present, the muzzlePoint node will be used instead.

**ambient sequence**
- Cyclic sequence to play while Image is mounted. This sequence is optional.

**spin sequence**
- Cyclic sequence to play while Image is mounted. This sequence is optional. See ShapeBaseImageSpinState.
Weapon Muzzle Flash

When the Image is used as a weapon, a sequence can be added to display a muzzle flash when the weapon is fired (if stateSequenceRandomFlash is set to true for the firing state). The name of the muzzle flash sequence is the same as the state sequence (eg. fire), but with '_vis' appended (eg. fire_vis).

In the example below, the muzzle flash is made up of three quads; one facing the player, and two crossed quads pointing out of the weapon so viewers perpendicular to the player will also see the flash.

The visibility of the muzzle flash mesh is animated on for 1 frame then off for 1 frame as shown below, but any Torque supported animation method could be used as well. For example, the node the quads are attached to could be rotated or scaled, or the mesh Material could be animated (UV or frame) to provide further variation.
First Person Shape [Optional]

The ShapeBaseImageData supports an optional shape that is displayed when the player is in a first person view. This shape is defined using the shapeFileFP property. You also must set an eyeOffset or make use of an eye mount node for this shape to be used in a first person view.

Having this second shape defined provides for more flexibility between 3rd person (and what other players see) and 1st person views. In a typical first person shooter the 3rd person weapon is not as detailed and supports a limited number of animation sequences. Just enough for the other players in the game to get a sense of what the player is doing. Then the 1st person weapon has a lot more detail, such as moving parts, etc. It may also have some arms and hands included that are animated when reloading the weapon and other actions. Only the player holding the weapon sees all of this.

There are a number of things to keep in mind if you make use of shapeFileFP:

Requires eyeOffset or useEyeNode

In Torque a 1st person view of an Image is only active if an eyeOffset is defined, or you have the useEyeNode flag set. This is also true if you wish to make use of a specific first person shape.

Player First Person Rendering

You'll probably want to turn off the rendering of the player in first person. Otherwise its shape and animation may not match the first person Image

Animation Sequences and Naming

You are not required to have the same animation sequences defined for the standard Image shape and the 1st person Image shape. Any sequences that are missing are just ignored. However, if you wish to have the same sequence play for a particular Image state on both shapes you must use the same animation sequence name between them. Image states only support a single sequence name.
Animation Sequence Playback Speed

The playback speed of a state's animation sequence may be scaled to match the length of the state's timeout. The standard (3rd person) shape and 1st person shape may be set up with different playback scaling options using the stateScaleAnimation and stateScaleAnimationFP flags.

stateEmitterNode

This optional node's world position will not likely line up between the standard shape and the 1st person shape. If the player switches from 3rd person to 1st person (or vice versa) while particles are being emitted from this node, the emission point will appear to shift. This normally isn't an issue for most uses of this node, such as smoke from the barrel after the weapon has been fired. As these particles are a client-side only effect, no other players will see this shift as the standard shape will always be used.

Server Always In First Person

The server always operates in 1st person mode when it comes to transform calculations for an Image's nodes. This means that the server will use the shapeFileFP nodes when it is present. This is something to be aware of when building the two shape files.
Animation Sequence Transitions

Starting with T3D 1.2 control is now given over transitioning from one image state's sequence to another state's sequence. The new state "stateSequenceTransitionIn" and "stateSequenceTransitionOut" flags dictate if the current state's sequence should be transitioned into when changing to the current state, or transitioned out of when switching to a new state. However, there are times when you don't want to do an animation sequence transition regardless of which state you are coming from. An example of this is the traditional "Fire" state. A Fire state should play immediately and not be transitioned into. In these cases a state may set the "stateSequenceNeverTransition" flag. With that set a state's sequence will begin to play immediately.
Animation Sequence Selection

When it comes to choosing what sequence to play on the mounted image there are now some new rules. Under 1.1 when an image state requested a named sequence that is found on the mounted image and played -- its action sequence. This still occurs under 1.2. However, it is now possible to modify the name of the sequence to play based on some prefixes. PlayerData now has two additional, optional fields: imageAnimPrefix and imageAnimPrefixFP. Just like how these same fields on ShapeBaseImageData can modify when sequences are played on the player based on what is mounted (see Player class documentation), these two PlayerData fields can modify what sequence is played on the mounted image based on the mounting player. This becomes especially useful when combined with 1st person arms -- although here we're just talking about weapons/mounted images.

Let's suppose we have two types of player: Soldier and Alien. We may want each type of player to use the same weapon slightly differently (or even radically differently, such as the Alien holding the weapon upside down). We use the "Soldier" anim prefix in the soldier's datablock and the "Alien" prefix in the alien's datablock. Now when looking up the sequence for a weapon's fire state -- usually called "fire" by convention -- the appropriate prefix is added first. And if that prefixed sequence is not found, then we fall back to the standard sequence name. So the soldier's sequence name search looks like this:

- Soldier_fire
- fire

and the alien's sequence name search looks like this:

- Alien_fire
- fire

This gives the artist greater control over how the weapons look. And because there are separate prefixes possible on PlayerData for 1st person and 3rd person you can mix and match as appropriate. So
you could set a prefix for 1st person, but leave it blank for 3rd person (don't do anything special in 3rd person).

Another way that an image state's sequence name could be modified is through the new `ShapeBase::setImageScriptAnimPrefix()` method. This allows you to insert an additional prefix into the name look up. The current scripts pass along the player's current pose name, but anything could be passed in based on game play. This can be even more useful with the 1st person arms. You could then have a weapon idle state when swimming that moves the weapon (and 1st person arms) in a gentle swim motion. When you combine the `PlayerData` prefix with the script anim prefix and finally with the image state sequence name, you end up with the following sequence name search pattern (the first found is used):

- PlayerDataPrefix_ScriptAnimPrefix_StateSequenceName
- PlayerDataPrefix_StateSequenceName
- ScriptAnimPrefix_StateSequenceName
- StateSequenceName

Whenever `ShapeBase::setImageScriptAnimPrefix()` is called there is a transition from the currently playing state sequence into the new script prefixed animation sequence. In our example, this allows for a transition from walking to swimming for the weapon. The new `ShapeBaseImageData` `scriptAnimationTransitionTime` controls how long to take for this transition.
**eyeMount Node [Optional]**

As with 1.1 the placement of the 1st person image may be set with the eyeOffset parameter. Now with 1.2 the 1st person image may be placed based on a node in the 1st person DTS/DAE shape, the "eyeMount" node. When the ShapeBaseImageData's useEyeNode parameter is made true, the image is effectively mounted to the 3rd person player's "eye" node, locking it into place. This allows the artist in their 3D application to precisely place the 1st person weapon in view when their 3D app's camera is placed on the eyeMount node and has the same field of view as Torque. This is very handy when animating the 1st person weapon, especially with 1st person arms.

Also with 1.2 an image that is placed with the eyeMount node may have an "eye" node defined. When found the player's camera is mounted to the image's "eye" node rather than the 3rd person player's "eye" node. This allows for animating the camera such as during a fire sequence.

Allowing for this much control does have a potential down side. In order for a weapon to fire correctly on the server it needs to have its muzzle point at the correct location. If a weapon's root pose (without animation) doesn't have its muzzle point at roughly the same location as when the weapon is fired, then the new `ShapeBaseImageData animateOnServer` flag should be set. When set the server will perform all state machine animation to ensure the muzzle point is at the correct location when required. This puts an extra strain on the server. If care is taken when building the weapons such that the root pose is close enough to the fire pose, then you can safely leave the "animateOnServer" flag off and not have to worry about the extra server load.
Special State Triggers

Starting with 1.2 there are now a number of new triggers that may be set for a ShapeBaseImageData's state machine to react to. These provide greater game play control over an image's state flow. The first are the "stateTransitionOnMotion" and "stateTransitionOnNoMotion" triggers. This trigger occurs whenever the mounting ShapeBase (usually a Player) has x, y or z motion applied through the Move structure. From a Player perspective this means whenever the user moves their player forwards, backwards or strafes. That has been used to provide weapons a slight bobbing appearance (using an animation sequence) when the weapon is idle. Fire and Reload states don't usually make use of these triggers to keep those actions solid.

There has always been a target trigger for ShapeBaseImageData but under 1.1 it was not possible to set it, nor was it used. Starting with 1.2 you can now set the target trigger in script using ShapeBase::setImageTargetState() and use stateTransitionOnTarget and stateTransitionOnNoTarget for whatever game play reasons are required.

Finally, there are four new generic triggers that may be set from script and used for whatever purpose the game play imposes. These are "stateTransitionGeneric0In", "stateTransitionGeneric1In", etc. and "stateTransitionGeneric0Out", "stateTransition1Out" etc. The FPS Tutorial weapons use the first generic trigger to indicate that the player is sprinting and switch to a Sprint state to prevent firing of the weapon. Other possible uses are for iron sights.
Special States

The client and server move through a ShapeBaseImageData's state machine independantly according to various triggers, timeouts, etc. The client is not normally told to move to a specific state when the server does. However, there are three instances where the client is told by the server to immediately jump to a given state. This ensures that the client's experience matches the server at key moments. As such, only one of each of these states may exist in a single ShapeBaseImageData state machine at a time.

The fire state is the first such state. It is indicated by setting the state's "stateFire" flag to true. This is the state immediately jumped to when the weapon begins to fire.

The alternate fire state is the second forced jump point (new in 1.2). It is indicated by setting the state's "stateAlternateFire" flag to true. Not all weapons have an alternate fire state. In fact most games treat a weapon's alternate fire as a separate weapon altogether.

The reload state is the last special state (new in 1.2). It is indicated by setting the state's "stateReload" flag to true. This state is triggered if the weapon makes use of the new 1.2 ammo clip system and the weapon is reloading a clip, either automatically or manually triggered by the client.
Member Function Documentation

```cpp
void ShapeBaseImageData::onMount (ShapeBase obj, 
    int slot, 
    float dt
)
```

Called when the Image is first mounted to the object.

**Parameters:**

- `obj` object that this Image has been mounted to
- `slot` Image mount slot on the object
- `dt` time remaining in this Image update

```cpp
void ShapeBaseImageData::onUnmount (ShapeBase obj, 
    int slot, 
    float dt
)
```

Called when the Image is unmounted from the object.

**Parameters:**

- `obj` object that this Image has been unmounted from
- `slot` Image mount slot on the object
- `dt` time remaining in this Image update
Member Data Documentation

bool ShapeBaseImageData::accuFire

Flag to control whether the Image's aim is automatically converged with the crosshair.

Currently unused.

bool ShapeBaseImageData::animateAllShapes

Indicates that all shapes should be animated in sync.

When multiple shapes are defined for this image datablock, each of them are automatically animated in step with each other. This allows for easy switching between between shapes when some other condition changes, such as going from first person to third person, and keeping their look consistent. If you know that you'll never switch between shapes on the fly, such as players only being allowed in a first person view, then you could set this to false to save some calculations.

There are other circumstances internal to the engine that determine that only the current shape should be animated rather than all defined shapes. In those cases, this property is ignored.

Note:

This property is only important if you have more than one shape defined, such as shapeFileFP.

See also:

shapeFileFP

bool ShapeBaseImageData::animateOnServer
Indicates that the image should be animated on the server.

In most cases you'll want this set if you're using useEyeNode. You may also want to set this if the muzzlePoint is animated while it shoots. You can set this to false even if these previous cases are true if the image's shape is set up in the correct position and orientation in the 'root' pose and none of the nodes are animated at key times, such as the muzzlePoint essentially remaining at the same position at the start of the fire state (it could animate just fine after the projectile is away as the muzzle vector is only calculated at the start of the state).

You'll also want to set this to true if you're animating the camera using the image's 'eye' node -- unless the movement is very subtle and doesn't need to be reflected on the server.

**Note:**

Setting this to true causes up to four animation threads to be advanced on the server for each instance in use, although for most images only one or two are actually defined.

**See also:**

useEyeNode

---

**Point3F ShapeBaseImageData::camShakeAmp**

Amplitude of the camera shaking effect.

**See also:**

shakeCamera

---

**Point3F ShapeBaseImageData::camShakeFreq**

Frequency of the camera shaking effect.
See also:

shakeCamera

DebrisData ShapeBaseImageData::casing

DebrisData datablock to use for ejected casings.

See also:

stateEjectShell

bool ShapeBaseImageData::cloakable

Whether this Image can be cloaked.
Currently unused.

bool ShapeBaseImageData::computeCRC

If true, verify that the CRC of the client's Image matches the server's CRC for the Image when loaded by the client.

bool ShapeBaseImageData::correctMuzzleVector

Flag to adjust the aiming vector to the eye's LOS point.
Flag to adjust the aiming vector to the eye's LOS point when in 1st person view.

See also:

ShapeBase::getMuzzleVector()
**bool ShapeBaseImageData::correctMuzzleVectorTP**

Flag to adjust the aiming vector to the camera's LOS point when in 3rd person view.

**See also:**

ShapeBase::getMuzzleVector()

---

**bool ShapeBaseImageData::emap**

Whether to enable environment mapping on this Image.

---

**MatrixPosition ShapeBaseImageData::eyeOffset**

"X Y Z" translation offset from the ShapeBase model's eye node.

When in first person view, this is the offset from the eye node to place the gun. This gives the gun a fixed point in space, typical of a lot of FPS games.

**See also:**

eyeRotation

---

**MatrixRotation ShapeBaseImageData::eyeRotation**

"X Y Z ANGLE" rotation offset from the ShapeBase model's eye node.

When in first person view, this is the rotation from the eye node to place the gun.

**See also:**

eyeOffset
bool ShapeBaseImageData::firstPerson

Set to true to render the image in first person.

caseString ShapeBaseImageData::imageAnimPrefix

Passed along to the mounting shape to modify animation sequences played in third person. [optional].

caseString ShapeBaseImageData::imageAnimPrefixFP

Passed along to the mounting shape to modify animation sequences played in first person. [optional].

float ShapeBaseImageData::lightBrightness

Brightness of the light this Image emits.

Only valid for WeaponFireLight.

See also:
  lightType

ColorF ShapeBaseImageData::lightColor

The color of light this Image emits.

See also:
  lightType

int ShapeBaseImageData::lightDuration
Duration in SimTime of Pulsing and WeaponFire type lights.

**See also:**
- `lightType`

```plaintext
float ShapeBaseImageData::lightRadius
```

Radius of the light this Image emits.

**See also:**
- `lightType`

```plaintext
ShapeBaseImageLightType ShapeBaseImageData::lightType
```

The type of light this Image emits.

**See also:**
- `ShapeBaseImageLightType`

```plaintext
float ShapeBaseImageData::mass
```

Mass of this Image.

This is added to the total mass of the `ShapeBase` object.

```plaintext
int ShapeBaseImageData::maxConcurrentSounds
```

Maximum number of sounds this Image can play at a time.

Any value <= 0 indicates that it can play an infinite number of sounds.
**float** `ShapeBaseImageData::minEnergy`

Minimum Image energy for it to be operable.

**See also:**
- `usesEnergy`

**int** `ShapeBaseImageData::mountPoint`

Mount node # to mount this Image to.

This should correspond to a mount# node on the `ShapeBase` derived object we are mounting to.

**MatrixPosition** `ShapeBaseImageData::offset`

"X Y Z" translation offset from this Image's `mountPoint` node to attach to.

Defaults to "0 0 0". ie. attach this Image's `mountPoint` node to the `ShapeBase` model's mount# node without any offset.

**See also:**
- `rotation`

**ProjectileData** `ShapeBaseImageData::Projectile`

The projectile fired by this Image.

**MatrixRotation** `ShapeBaseImageData::rotation`
"X Y Z ANGLE" rotation offset from this Image's `mountPoint` node to attach to.

Defaults to "0 0 0". ie. attach this Image's `mountPoint` node to the ShapeBase model's mount# node without any additional rotation.

**See also:**

offset

---

**float ShapeBaseImageData::scriptAnimTransitionTime**

The amount of time to transition between the previous sequence and new sequence when the script prefix has changed.

When setImageScriptAnimPrefix() is used on a ShapeBase that has this image mounted, the image will attempt to switch to the new animation sequence based on the given script prefix. This is the amount of time it takes to transition from the previously playing animation sequence to the new script prefix-based animation sequence.

**See also:**

ShapeBase::setImageScriptAnimPrefix()

---

**bool ShapeBaseImageData::shakeCamera**

Flag indicating whether the camera should shake when this Image fires.

**Note:**

Camera shake only works properly if the player is in control of the one and only shapeBase object in the scene which fires an Image that uses camera shake.
filename **ShapeBaseImageData::shapeFile**

The DTS or DAE model to use for this Image.

filename **ShapeBaseImageData::shapeFileFP**

The DTS or DAE model to use for this Image when in first person.

This is an optional parameter that also requires either eyeOffset or useEyeNode to be set. If none of these conditions is met then shapeFile will be used for all cases.

Typically you set a first person image for a weapon that includes the player's arms attached to it for animating while firing, reloading, etc. This is typical of many FPS games.

**See also:**
- eyeOffset
- useEyeNode

**Point3F ShapeBaseImageData::shellExitDir**

Vector direction to eject shell casings.

**See also:**
- casing

**float ShapeBaseImageData::shellExitVariance**

Variance (in degrees) from the shellExitDir vector to eject casings.

**See also:**
- shellExitDir
**float ShapeBaseImageData::shellVelocity**

Speed at which to eject casings.

**See also:**

casing

**bool ShapeBaseImageData::stateAllowImageChange[31]**

If false, other Images will temporarily be blocked from mounting while the state machine is executing the tasks in this state.

For instance, if we have a rocket launcher, the player shouldn't be able to switch out *while* firing. So, you'd set stateAllowImageChange to false in firing states, and true the rest of the time.

**bool ShapeBaseImageData::stateAlternateFire[31]**

The first state with this set to true is the state entered by the client when it receives the 'altFire' event.

**bool ShapeBaseImageData::stateDirection[31]**

Direction of the animation to play in this state.

True is forward, false is backward.

**bool ShapeBaseImageData::stateEjectShell[31]**

If true, a shell casing will be ejected in this state.
ParticleEmitterData ShapeBaseImageData::stateEmitter

Emitter to generate particles in this state (from muzzle point or specified node).

See also:

stateEmitterNode

string ShapeBaseImageData::stateEmitterNode

Name of the node to emit particles from.

See also:

stateEmitter

float ShapeBaseImageData::stateEmitterTime

How long (in seconds) to emit particles on entry to this state.

float ShapeBaseImageData::stateEnergyDrain

Amount of energy to subtract from the Image in this state.

Energy is drained at stateEnergyDrain units/tick as long as we are in this state.

See also:

usesEnergy

bool ShapeBaseImageData::stateFire

The first state with this set to true is the state entered by the client
when it receives the 'fire' event.

bool ShapeBaseImageData::stateIgnoreLoadedForReady

If set to true, and both ready and loaded transitions are true, the ready transition will be taken instead of the loaded transition.

A state is 'ready' if pressing the fire trigger in that state would transition to the fire state.

ShapeBaseImageLoadedState ShapeBaseImageData::stateLoadedFlag

Set the loaded state of the Image.

- IgnoreLoaded: Don't change Image loaded state.
- Loaded: Set Image loaded state to true.
- NotLoaded: Set Image loaded state to false.

See also:

ShapeBaseImageLoadedState
caseString ShapeBaseImageData::stateName

Name of this state.

ShapeBaseImageRecoilState ShapeBaseImageData::stateRecoil

Type of recoil sequence to play on the ShapeBase object on entry to this state.

- NoRecoil: Do not play a recoil sequence.
- LightRecoil: Play the light_recoil sequence.
- MediumRecoil: Play the medium_recoil sequence.
- HeavyRecoil: Play the heavy_recoil sequence.

See also:
ShapeBaseImageRecoilState

```cpp
bool ShapeBaseImageData::stateReload[31]
```

The first state with this set to true is the state entered by the client when it receives the 'reload' event.

```cpp
bool ShapeBaseImageData::stateScaleAnimation[31]
```

If true, the timeScale of the stateSequence animation will be adjusted such that the sequence plays for stateTimeoutValue seconds.

```cpp
bool ShapeBaseImageData::stateScaleAnimationFP[31]
```

If true, the timeScale of the first person stateSequence animation will be adjusted such that the sequence plays for stateTimeoutValue seconds.

```cpp
bool ShapeBaseImageData::stateScaleShapeSequence[31]
```

Indicates if the sequence to be played on the mounting shape should be scaled to the length of the state.

```cpp
caseString ShapeBaseImageData::stateScript[31]
```

Method to execute on entering this state.
Scoped to this image class name, then \texttt{ShapeBaseImageData}. The script callback function takes the same arguments as the \texttt{onMount} callback.

**See also:**

\texttt{onMount()} for the same arguments as this callback.

\begin{Verbatim}
\begin{tabular}{|l|}
\hline
\textbf{string ShapeBaseImageData::stateSequence[31]} \\
Name of the sequence to play on entry to this state. \\
\hline
\textbf{bool ShapeBaseImageData::stateSequenceNeverTransition[31]} \\
Never allow a transition to this sequence. Often used for a fire sequence. \\
\hline
\textbf{bool ShapeBaseImageData::stateSequenceRandomFlash[31]} \\
If true, the muzzle flash sequence will be played while in this state. \\
The name of the muzzle flash sequence is the same as \texttt{stateSequence}, with "\_vis" at the end. \\
\hline
\textbf{bool ShapeBaseImageData::stateSequenceTransitionIn[31]} \\
Do we transition to the state's sequence when we enter the state? \\
\hline
\textbf{bool ShapeBaseImageData::stateSequenceTransitionOut[31]} \\
Do we transition to the new state's sequence when we leave the state? \\
\hline
\end{tabular}
\end{Verbatim}
**float ShapeBaseImageData::stateSequenceTransitionTime[31]**

The time to transition in or out of a sequence.

**string ShapeBaseImageData::stateShapeSequence[31]**

Name of the sequence that is played on the mounting shape.

**SFXTrack ShapeBaseImageData::stateSound[31]**

Sound to play on entry to this state.

**ShapeBaseImageSpinState ShapeBaseImageData::stateSpinThread[31]**

Controls how fast the 'spin' animation sequence will be played in this state.

- Ignore: No change to the spin sequence.
- Stop: Stops the spin sequence at its current position.
- SpinUp: Increase spin sequence timeScale from 0 (on state entry) to 1 (after stateTimeoutValue seconds).
- SpinDown: Decrease spin sequence timeScale from 1 (on state entry) to 0 (after stateTimeoutValue seconds).
- FullSpeed: Resume the spin sequence playback at its current position with timeScale=1.

**See also:**

  ShapeBaseImageSpinState

**float ShapeBaseImageData::stateTimeoutValue[31]**
Time in seconds to wait before transitioning to stateTransitionOnTimeout.

**string** `ShapeBaseImageData::stateTransitionGeneric0In[31]`

Name of the state to transition to when the generic trigger 0 state changes to true.

**string** `ShapeBaseImageData::stateTransitionGeneric0Out[31]`

Name of the state to transition to when the generic trigger 0 state changes to false.

**string** `ShapeBaseImageData::stateTransitionGeneric1In[31]`

Name of the state to transition to when the generic trigger 1 state changes to true.

**string** `ShapeBaseImageData::stateTransitionGeneric1Out[31]`

Name of the state to transition to when the generic trigger 1 state changes to false.

**string** `ShapeBaseImageData::stateTransitionGeneric2In[31]`

Name of the state to transition to when the generic trigger 2 state changes to true.

**string** `ShapeBaseImageData::stateTransitionGeneric2Out[31]`
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string ShapeBaseImageData::stateTransitionGeneric3In[31]</td>
<td>Name of the state to transition to when the generic trigger 3 state changes to true.</td>
</tr>
<tr>
<td>string ShapeBaseImageData::stateTransitionGeneric3Out[31]</td>
<td>Name of the state to transition to when the generic trigger 3 state changes to false.</td>
</tr>
<tr>
<td>string ShapeBaseImageData::stateTransitionOnAltTriggerDown[31]</td>
<td>Name of the state to transition to when the alt trigger state of the Image changes to false (alt fire button up).</td>
</tr>
<tr>
<td>string ShapeBaseImageData::stateTransitionOnAltTriggerUp[31]</td>
<td>Name of the state to transition to when the alt trigger state of the Image changes to true (alt fire button down).</td>
</tr>
<tr>
<td>string ShapeBaseImageData::stateTransitionOnAmmo[31]</td>
<td>Name of the state to transition to when the ammo state of the Image changes to true.</td>
</tr>
<tr>
<td>string ShapeBaseImageData::stateTransitionOnLoaded[31]</td>
<td>Name of the state to transition to when the generic trigger 2 state changes to false.</td>
</tr>
</tbody>
</table>
Name of the state to transition to when the loaded state of the Image changes to 'Loaded'.

```c++
string ShapeBaseImageData::stateTransitionOnMotion[31]
```

Name of the state to transition to when the Player moves.

```c++
string ShapeBaseImageData::stateTransitionOnNoAmmo[31]
```

Name of the state to transition to when the ammo state of the Image changes to false.

```c++
string ShapeBaseImageData::stateTransitionOnNoMotion[31]
```

Name of the state to transition to when the Player stops moving.

```c++
string ShapeBaseImageData::stateTransitionOnNoTarget[31]
```

Name of the state to transition to when the Image loses a target.

```c++
string ShapeBaseImageData::stateTransitionOnNotLoaded[31]
```

Name of the state to transition to when the loaded state of the Image changes to 'Empty'.

```c++
string ShapeBaseImageData::stateTransitionOnNotWet[31]
```

Name of the state to transition to when the Image exits the water.
string ShapeBaseImageData::stateTransitionOnTarget

Name of the state to transition to when the Image gains a target.

string ShapeBaseImageData::stateTransitionOnTimeout

Name of the state to transition to when we have been in this state for stateTimeoutValue seconds.

string ShapeBaseImageData::stateTransitionOnTriggerDown

Name of the state to transition to when the trigger state of the Image changes to false (fire button released).

string ShapeBaseImageData::stateTransitionOnTriggerUp

Name of the state to transition to when the trigger state of the Image changes to true (fire button down).

string ShapeBaseImageData::stateTransitionOnWet

Name of the state to transition to when the Image enters the water.

bool ShapeBaseImageData::stateWaitForTimeout

If false, this state ignores stateTimeoutValue and transitions immediately if other transition conditions are met.

bool ShapeBaseImageData::useEyeNode
Mount image using image's eyeMount node and place the camera at the image's eye node (or at the eyeMount node if the eye node is missing).

When in first person view, if an 'eyeMount' node is present in the image's shape, this indicates that the image should mount eyeMount node to Player eye node for image placement. The Player's camera should also mount to the image's eye node to inherit any animation (or the eyeMount node if the image doesn't have an eye node).

**Note:**

Used instead of eyeOffset.
Read about the animateOnServer field as you may want to set it to true if you're using useEyeNode.

**See also:**

- eyeOffset
- animateOnServer

```cpp
bool ShapeBaseImageData::useRemainderDT
```

If true, allow multiple timeout transitions to occur within a single tick (useful if states have a very small timeout).

```cpp
bool ShapeBaseImageData::usesEnergy
```

Flag indicating whether this Image uses energy instead of ammo. The energy level comes from the ShapeBase object we're mounted to.

**See also:**

- ShapeBase::setEnergyLevel()
SimDataBlock Class Reference

Inheritance diagram for SimDataBlock:
List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>void reloadOnLocalClient ()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reload the datablock. This can only be used with a local client configuration.</td>
</tr>
</tbody>
</table>
Detailed Description

Datablocks and Networking

Client-Side Datablocks
Member Function Documentation

```cpp
void SimDataBlock::reloadOnLocalClient()
```

Reload the datablock. This can only be used with a local client configuration.
**SimGroup Class Reference**

[Console, Scripting]

A collection of SimObjects that are owned by the group. More...

Inheritance diagram for SimGroup:
List of all members.
Detailed Description

A collection of SimObjects that are owned by the group.

A SimGroup is a stricter form of SimSet. SimObjects may only be a member of a single SimGroup at a time. The SimGroup will automatically enforce the single-group-membership rule (i.e., adding an object to a SimGroup will cause it to be removed from its current SimGroup, if any).

Deleting a SimGroup will also delete all SimObjects in the SimGroup.

Example:

```csharp
// Create a SimGroup for particle emitters
new SimGroup(Emitters) {
    canSaveDynamicFields = "1";

    new ParticleEmitterNode(CrystalEmmiter) {
        active = "1";
        emitter = "dustEmitter";
        velocity = "1";
        dataBlock = "GenericSmokeEmitterNode";
        position = "-61.6276 2.1142 4.45027"
        rotation = "1 0 0 0"
        scale = "1 1 1"
        canSaveDynamicFields = "1"
    }

    new ParticleEmitterNode(Steam1) {
        active = "1";
        emitter = "SlowSteamEmitter";
        velocity = "1";
        dataBlock = "GenericSmokeEmitterNode"
    }
}
```
position = "-25.0458 1.55289 2.51308"
rotation = "1 0 0 0"
scale = "1 1 1"
canSaveDynamicFields = "1";


SimObject Class Reference
[Console, Game]

Base class for almost all objects involved in the simulation. More...

Inheritance diagram for SimObject:
List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>void assignFieldsFrom (SimObject fromObject)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Copy fields from another object onto this one. The objects must be of same type. Everything from the object will overwrite what's in this object; extra fields in this object will remain. This includes dynamic fields.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void assignPersistentId ()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assign a persistent ID to the object if it does not already have one.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string call (string method, string args...)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamically call a method on an object.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SimObject clone ()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a copy of this object.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SimObject deepClone ()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a copy of this object and all its subobjects.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void delete ()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete and remove the object.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void dump (bool detailed=false)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dump a description of all fields and methods defined on this object to the console.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void dumpClassHierarchy ()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dump the native C++ class hierarchy of this object's C++ class to the console.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void dumpGroupHierarchy ()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dump the hierarchy of this object up to RootGroup to the console.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ArrayObject dumpMethods ()</th>
</tr>
</thead>
<tbody>
<tr>
<td>List the methods defined on this object.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool getCanSave ()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get whether the object will be included in saves.</td>
</tr>
<tr>
<td>Function</td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td><strong>string</strong> <code>getClassName()</code></td>
</tr>
<tr>
<td><strong>string</strong> <code>getClassNamespace()</code></td>
</tr>
<tr>
<td><strong>ArrayOfObject</strong> <code>getDebugInfo()</code></td>
</tr>
<tr>
<td><strong>int</strong> <code>getDeclarationLine()</code></td>
</tr>
<tr>
<td><strong>string</strong> <code>getDynamicField(int index)</code></td>
</tr>
<tr>
<td><strong>int</strong> <code>getDynamicFieldCount()</code></td>
</tr>
<tr>
<td><strong>string</strong> <code>getField(int index)</code></td>
</tr>
<tr>
<td><strong>int</strong> <code>getFieldCount()</code></td>
</tr>
<tr>
<td><strong>string</strong> <code>getFieldType(string fieldName)</code></td>
</tr>
<tr>
<td><strong>string</strong> <code>getFieldValue(string fieldName, int index=-1)</code></td>
</tr>
<tr>
<td><strong>string</strong> <code>getFilename()</code></td>
</tr>
<tr>
<td><strong>SimGroup</strong> <code>getGroup()</code></td>
</tr>
<tr>
<td><strong>int</strong> <code>getId()</code></td>
</tr>
<tr>
<td>Function</td>
</tr>
<tr>
<td>---------------------------</td>
</tr>
<tr>
<td>string getInternalName ()</td>
</tr>
<tr>
<td>string getName ()</td>
</tr>
<tr>
<td>string getSuperClassNamespace ()</td>
</tr>
<tr>
<td>bool isChildOfGroup (SimGroup group)</td>
</tr>
<tr>
<td>bool isEditorOnly ()</td>
</tr>
<tr>
<td>bool isExpanded ()</td>
</tr>
<tr>
<td>bool isField (string fieldName)</td>
</tr>
<tr>
<td>bool isInNamespaceHierarchy (string name)</td>
</tr>
<tr>
<td>bool isMemberOfClass (string className)</td>
</tr>
<tr>
<td>bool isMethod (string methodName)</td>
</tr>
<tr>
<td>bool isNameChangeAllowed ()</td>
</tr>
<tr>
<td>bool isSelected ()</td>
</tr>
<tr>
<td>bool save (string fileName, bool selectedOnly=false)</td>
</tr>
<tr>
<td>Function</td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td><code>Save</code></td>
</tr>
<tr>
<td><code>int schedule (float time, string method, string args...)</code></td>
</tr>
<tr>
<td><code>void setCanSave (bool value=true)</code></td>
</tr>
<tr>
<td><code>void setClassNamespace (string name)</code></td>
</tr>
<tr>
<td><code>void setEditorOnly (bool value=true)</code></td>
</tr>
<tr>
<td><code>void setFieldType (string fieldName, string type)</code></td>
</tr>
<tr>
<td><code>bool setFieldValue (string fieldName, string value, int index=-1)</code></td>
</tr>
<tr>
<td><code>void setFilename (string fileName)</code></td>
</tr>
<tr>
<td><code>void setHidden (bool value=true)</code></td>
</tr>
<tr>
<td><code>void setInternalName (string newInternalName)</code></td>
</tr>
<tr>
<td><code>void setIsExpanded (bool state=true)</code></td>
</tr>
<tr>
<td><code>void setIsSelected (bool state=true)</code></td>
</tr>
<tr>
<td><code>void setLocked (bool value=true)</code></td>
</tr>
<tr>
<td><code>void setName (string newName)</code></td>
</tr>
<tr>
<td>void</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>
# Public Attributes

## Persistence

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>canSave</td>
<td>Whether the object can be saved out. If false, the object is purely transient in nature.</td>
</tr>
<tr>
<td>bool</td>
<td>canSaveDynamicFields</td>
<td>True if dynamic fields (added at runtime) should be saved. Defaults to true.</td>
</tr>
<tr>
<td>pid</td>
<td>persistentId</td>
<td>The universally unique identifier for the object.</td>
</tr>
</tbody>
</table>

## Object

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>class</td>
<td>Script class of object.</td>
</tr>
<tr>
<td>string</td>
<td>className</td>
<td>Script class of object.</td>
</tr>
<tr>
<td>string</td>
<td>internalName</td>
<td>Optional name that may be used to lookup this object within a SimSet.</td>
</tr>
</tbody>
</table>

## SimObject

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>superClass</td>
<td>Script super-class of object.</td>
</tr>
</tbody>
</table>

## Editing

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>hidden</td>
<td>Whether the object is visible.</td>
</tr>
<tr>
<td>bool</td>
<td>locked</td>
<td></td>
</tr>
</tbody>
</table>
Whether the object can be edited.

Ungrouped

<table>
<thead>
<tr>
<th>string</th>
<th>name</th>
</tr>
</thead>
</table>
|         | Optional global name of this object.
Detailed Description

Base class for almost all objects involved in the simulation.
Introduction

SimObject is a base class for most of the classes you'll encounter working in Torque. It provides fundamental services allowing "smart" object referencing, creation, destruction, organization, and location. Along with SimEvent, it gives you a flexible event-scheduling system, as well as laying the foundation for the in-game editors, GUI system, and other vital subsystems.
Subclassing

You will spend a lot of your time in Torque subclassing, or working with subclasses of, `SimObject`. `SimObject` is designed to be easy to subclass.

You should not need to override anything in a subclass except:

- The constructor/destructor.
- `processArguments()`
- `onAdd()`/`onRemove()`
- `onGroupAdd()`/`onGroupRemove()`
- `onNameChange()`
- `onStaticModified()`
- `onDeleteNotify()`
- `onEditorEnable()`/`onEditorDisable()`
- `inspectPreApply()`/`inspectPostApply()`
- things from `ConsoleObject` (see `ConsoleObject` docs for specifics)

Of course, if you know what you're doing, go nuts! But in most cases, you shouldn't need to touch things not on that list.

When you subclass, you should define a typedef in the class, called `Parent`, that references the class you're inheriting from.

```cpp
class mySubClass : public SimObject {
    typedef SimObject Parent;
    ...
}
```

Then, when you override a method, put in:

```cpp
bool mySubClass::onAdd()
{
    if(!Parent::onAdd())
        return false;
}
```
Of course, you want to replace onAdd with the appropriate method call.
A SimObject's Life Cycle

SimObjects do not live apart. One of the primary benefits of using a SimObject is that you can uniquely identify it and easily find it (using its ID). Torque does this by keeping a global hierarchy of SimGroups - a tree - containing every registered SimObject. You can then query for a given object using Sim::findObject() (or SimSet::findObject() if you want to search only a specific set).

```
// Three examples of registering an

// Method 1:
AIClient *aiPlayer = new AIClient();
aiPlayer->registerObject();

// Method 2:
ActionMap* globalMap = new ActionMap;
  globalMap->registerObject("GlobalActionMap");

// Method 3:
bool reg = mObj->registerObject(id)
```

Registering a SimObject performs these tasks:

- Marks the object as not cleared and not removed.
- Assigns the object a unique SimObjectID if it does not have one already.
- Adds the object to the global name and ID dictionaries so it can be found again.
- Calls the object's onAdd() method. **Note**: SimObject::onAdd() performs some important initialization steps. See [here](#) for details" on how to properly subclass SimObject.
- If onAdd() fails (returns false), it calls unregisterObject().
- Checks to make sure that the SimObject was properly initialized (and asserts if not).
Calling `registerObject()` and passing an ID or a name will cause the object to be assigned that name and/or ID before it is registered.

Congratulations, you have now registered your object! What now?

Well, hopefully, the `SimObject` will have a long, useful life. But eventually, it must die.

There are a two ways a `SimObject` can die.

- First, the game can be shut down. This causes the root `SimGroup` to be unregistered and deleted. When a `SimGroup` is unregistered, it unregisters all of its member `SimObjects`; this results in everything that has been registered with Sim being unregistered, as everything registered with Sim is in the root group.
- Second, you can manually kill it off, either by calling `unregisterObject()` or by calling `deleteObject()`.

When you unregister a `SimObject`, the following tasks are performed:

- The object is flagged as removed.
- Notifications are cleaned up.
- If the object is in a group, then it removes itself from the group.
- Delete notifications are sent out.
- Finally, the object removes itself from the Sim globals, and tells Sim to get rid of any pending events for it.

If you call `deleteObject()`, all of the above tasks are performed, in addition to some sanity checking to make sure the object was previously added properly, and isn't in the process of being deleted. After the object is unregistered, it deallocates itself.
Torque Editors

SimObjects are one of the building blocks for the in-game editors. They provide a basic interface for the editor to be able to list the fields of the object, update them safely and reliably, and inform the object things have changed.

This interface is implemented in the following areas:

- `onNameChange()` is called when the object is renamed.
- `onStaticModified()` is called whenever a static field is modified.
- `inspectPreApply()` is called before the object's fields are updated, when changes are being applied.
- `inspectPostApply()` is called after the object's fields are updated.
- `onEditorEnable()` is called whenever an editor is enabled (for instance, when you hit F11 to bring up the world editor).
- `onEditorDisable()` is called whenever the editor is disabled (for instance, when you hit F11 again to close the world editor).

(Note: you can check the variable `gEditingMission` to see if the mission editor is running; if so, you may want to render special indicators. For instance, the `fxFoliageReplicator` renders inner and outer radii when the mission editor is running.)
The Console

*SimObject* extends ConsoleObject by allowing you to set arbitrary dynamic fields on the object, as well as statically defined fields. This is done through two methods, `setDataField` and `getDataField`, which deal with the complexities of allowing access to two different types of object fields.

Static fields take priority over dynamic fields. This is to be expected, as the role of dynamic fields is to allow data to be stored in addition to the predefined fields.

The fields in a *SimObject* are like properties (or fields) in a class.

Some fields may be arrays, which is what the array parameter is for; if it's non-null, then it is parsed with `dAtol` and used as an index into the array. If you access something as an array which isn't, then you get an empty string.

You don't need to read any further than this. Right now, `set/getDataField` are called a total of 6 times through the entire Torque codebase. Therefore, you probably don't need to be familiar with the details of accessing them. You may want to look at `Con::setData` instead. Most of the time you will probably be accessing fields directly, or using the scripting language, which in either case means you don't need to do anything special.

The functions to get/set these fields are very straightforward:

```
setDataField(StringTable->insert("locked", curObject->setDataField(curField, curFieldArray, STR.getStringValue());
```

For advanced users: There are two flags which control the behavior of these functions. The first is `ModStaticFields`, which controls whether or not the `DataField` functions look through the static fields (defined with `addField`; see `ConsoleObject` for details) of the class. The second is `ModDynamicFields`, which controls dynamically
defined fields. They are set automatically by the console constructor code.

See also:

SceneObject
Member Function Documentation

void SimObject::assignFieldsFrom (SimObject fromObject )

Copy fields from another object onto this one. The objects must be of same type. Everything from the object will overwrite what's in this object; extra fields in this object will remain. This includes dynamic fields.

Parameters:

fromObject The object from which to copy fields.

void SimObject::assignPersistentId ( )

Assign a persistent ID to the object if it does not already have one.

string SimObject::call (string method, string args... )

Dynamically call a method on an object.

Parameters:

method Name of method to call.
args Zero or more arguments for the method.

Returns:

The result of the method call.

SimObject SimObject::clone ( )

Create a copy of this object.
Returns:

An exact duplicate of this object.

SimObject SimObject::deepClone(
)

Create a copy of this object and all its subobjects.

Returns:

An exact duplicate of this object and all objects it references.

void SimObject::delete(
)

Delete and remove the object.

void SimObject::dump(bool detailed = false)

Dump a description of all fields and methods defined on this object to the console.

Parameters:

\textit{detailed} Whether to print detailed information about members.

void SimObject::dumpClassHierarchy(
)

Dump the native C++ class hierarchy of this object's C++ class to the console.

void SimObject::dumpGroupHierarchy(
)

Dump the hierarchy of this object up to RootGroup to the console.
ArrayObject SimObject::dumpMethods( )

List the methods defined on this object.

Each description is a newline-separated vector with the following elements:

- Minimum number of arguments.
- Maximum number of arguments.
- Prototype string.
- Full script file path (if script method).
- Line number of method definition in script (if script method).
- Documentation string (not including prototype). This takes up the remainder of the vector.

**Returns:**

An ArrayObject populated with (name,description) pairs of all methods defined on the object.

bool SimObject::getCanSave( )

Get whether the object will be included in saves.

**Returns:**

True if the object will be saved; false otherwise.

string SimObject::getClassName( )

Get the name of the C++ class which the object is an instance of.

**Returns:**

The name of the C++ class of the object.
string SimObject::getClassNamespace()

Get the name of the class namespace assigned to this object.

**Returns:**
The name of the 'class' namespace.

ArrayObject SimObject::getDebugInfo()

Return some behind-the-scenes information on the object.

**Returns:**
An ArrayObject filled with internal information about the object.

int SimObject::getDeclarationLine()

Get the line number at which the object is defined in its file.

**Returns:**
The line number of the object's definition in script.

**See also:**
getFilename()

string SimObject::getDynamicField(int index)

Get a value of a dynamic field by index.

**Parameters:**

index The index of the dynamic field.

**Returns:**
The value of the dynamic field at the given index or "".

```cpp
int SimObject::getDynamicFieldCount()
```

Get the number of dynamic fields defined on the object.

**Returns:**

The number of dynamic fields defined on the object.

```cpp
string SimObject::getField(int index)
```

Retrieve the value of a static field by index.

**Parameters:**

`index` The index of the static field.

**Returns:**

The value of the static field with the given index or "".

```cpp
int SimObject::getFieldCount()
```

Get the number of static fields on the object.

**Returns:**

The number of static fields defined on the object.

```cpp
string SimObject::getFieldType(string fieldName)
```

Get the console type code of the given field.

**Returns:**

The numeric type code for the underlying console type of the
given field.

```cpp
string SimObject::getFieldValue(string fieldName, int index = -1)
```

Return the value of the given field on this object.

**Parameters:**

- **fieldName**  
The name of the field. If it includes a field index, the index is parsed out.
- **index**  
Optional parameter to specify the index of an array field separately.

**Returns:**

- The value of the given field or "" if undefined.

```cpp
string SimObject::getFilename()
```

Returns the filename the object is attached to.

**Returns:**

- The name of the file the object is associated with; usually the file the object was loaded from.

Reimplemented in [CubemapData](#).

```cpp
SimGroup SimObject::getGroup()
```

Get the group that this object is contained in.

**Note:**

- If not assigned to particular [SimGroup](#), an object belongs to
Returns:
The SimGroup object to which the object belongs.

int SimObject::getId()

Get the underlying unique numeric ID of the object.

Note:
Object IDs are unique only during single engine runs.

Returns:
The unique numeric ID of the object.

string SimObject::getInternalName()

Get the internal name of the object.

Returns:
The internal name of the object.

string SimObject::getName()

Get the global name of the object.

Returns:
The global name assigned to the object.

string SimObject::getSuperClassNamespace()

Get the name of the superclass namespace assigned to this
Returns:
The name of the 'superClass' namespace.

bool SimObject::isChildOfGroup(SimGroup group)

Test whether the object belongs directly or indirectly to the given group.

Parameters:
  group The SimGroup object.

Returns:
  True if the object is a child of the given group or a child of a group that the given group is directly or indirectly a child to.

bool SimObject::isEditorOnly()

Return true if the object is only used by the editor.

Returns:
  True if this object exists only for the sake of editing.

bool SimObject::isExpanded()

Get whether the object has been marked as expanded. (in editor).

Returns:
  True if the object is marked expanded.

Reimplemented in GuiRolloutCtrl.
### bool SimObject::isField (string fieldName )

Test whether the given field is defined on this object.

**Parameters:**

- `fieldName` The name of the field.

**Returns:**

True if the object implements the given field.

### bool SimObject::isInNamespaceHierarchy (string name )

Test whether the namespace of this object is a direct or indirect child to the given namespace.

**Parameters:**

- `name` The name of a namespace.

**Returns:**

True if the given namespace name is within the namespace hierarchy of this object.

### bool SimObject::isMemberOfClass (string className )

Test whether this object is a member of the specified class.

**Parameters:**

- `className` Name of a native C++ class.

**Returns:**

True if this object is an instance of the given C++ class or any of its super classes.
bool SimObject::isMethod ( string methodName )

Test whether the given method is defined on this object.

**Parameters:**

*The* name of the method.

**Returns:**

True if the object implements the given method.

bool SimObject::isNameChangeAllowed ( )

Get whether this object may be renamed.

**Returns:**

True if this object can be renamed; false otherwise.

bool SimObject::isSelected ( )

Get whether the object has been marked as selected. (in editor).

**Returns:**

True if the object is currently selected.

bool SimObject::save ( string fileName, 
                         bool  selectedOnly = false, 
                         string preAppendString = "" 
                     )

Save out the object to the given file.

**Parameters:**

*fileName* 

The name of the file to save to.
selectedOnly If true, only objects marked as selected will be saved out.

preAppendString Text which will be prepended directly to the object serialization.

True on success, false on failure.

int SimObject::schedule(float time,
                        string method,
                        string args...)

Delay an invocation of a method.

Parameters:

  time The number of milliseconds after which to invoke the method. This is a soft limit.
  method The method to call.
  args The arguments with which to call the method.

Returns:

The numeric ID of the created schedule. Can be used to cancel the call.

void SimObject::setCanSave(bool value = true)

Set whether the object will be included in saves.

Parameters:

  value If true, the object will be included in saves; if false, it will be excluded.

void SimObject::setClassNamespace(string name)
Assign a class namespace to this object.

**Parameters:**

- **name** The name of the 'class' namespace for this object.

```cpp
void SimObject::setEditorOnly (bool value = true )
```

Set/clear the editor-only flag on this object.

**Parameters:**

- **value** If true, the object is marked as existing only for the editor.

```cpp
void SimObject::setFieldType (string fieldName, 
                               string type )
```

Set the console type code for the given field.

**Parameters:**

- **fieldName** The name of the dynamic field to change to type for.
- **type** The name of the console type.

**Note:**

This only works for dynamic fields. Types of static fields cannot be changed.

```cpp
bool SimObject::setFieldValue (string fieldName, 
                               string value, 
                               int index = -1 )
```
Set the value of the given field on this object.

**Parameters:**

- `fieldName` The name of the field to assign to. If it includes an array index, the index will be parsed out.
- `value` The new value to assign to the field.
- `index` Optional argument to specify an index for an array field.

**Returns:**
True.

```cpp
void SimObject::setFilename (string fileName )
```

Sets the object’s file name and path.

**Parameters:**

- `fileName` The name of the file to associate this object with.

```cpp
void SimObject::setHidden (bool value = true )
```

Hide/unhide the object.

**Parameters:**

- `value` If true, the object will be hidden; if false, the object will be unhidden.

Reimplemented in `ShapeBase`.

```cpp
void SimObject::setInternalName (string newInternalName )
```

Set the internal name of the object.
Parameters:

*newInternalName* The new internal name for the object.

```cpp
void SimObject::setIsExpanded(bool state = true )
```

Set whether the object has been marked as expanded. (in editor).

**Parameters:**

*state* True if the object is to be marked expanded; false if not.

```cpp
void SimObject::setIsSelected(bool state = true )
```

Set whether the object has been marked as selected. (in editor).

**Parameters:**

*state* True if object is to be marked selected; false if not.

```cpp
void SimObject::setLocked(bool value = true )
```

Lock/unlock the object in the editor.

**Parameters:**

*value* If true, the object will be locked; if false, the object will be unlocked.

```cpp
void SimObject::setName(string newName )
```

Set the global name of the object.

**Parameters:**

*newName* The new global name to assign to the object.
Note:
If name changing is disallowed on the object, the method will fail with a console error.

```cpp
void SimObject::setNameChangeAllowed(bool value = true)
```

Set whether this object can be renamed from its first name.

**Parameters:**
- `value` If true, renaming is allowed for this object; if false, trying to change the name of the object will generate a console error.

```cpp
void SimObject::setSuperClassNamespace(string name)
```

Assign a superclass namespace to this object.

**Parameters:**
- `name` The name of the 'superClass' namespace for this object.
<table>
<thead>
<tr>
<th>Member Data Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>bool SimObject::canSave</strong></td>
</tr>
<tr>
<td>Whether the object can be saved out. If false, the object is purely transient in nature.</td>
</tr>
<tr>
<td><strong>bool SimObject::canSaveDynamicFields</strong></td>
</tr>
<tr>
<td>True if dynamic fields (added at runtime) should be saved. Defaults to true.</td>
</tr>
<tr>
<td><strong>string SimObject::class</strong></td>
</tr>
<tr>
<td>Script class of object.</td>
</tr>
<tr>
<td><strong>string SimObject::className</strong></td>
</tr>
<tr>
<td>Script class of object.</td>
</tr>
<tr>
<td><strong>bool SimObject::hidden</strong></td>
</tr>
<tr>
<td>Whether the object is visible.</td>
</tr>
<tr>
<td><strong>string SimObject::internalName</strong></td>
</tr>
<tr>
<td>Optional name that may be used to lookup this object within a SimSet.</td>
</tr>
<tr>
<td>Variable</td>
</tr>
<tr>
<td>-------------------</td>
</tr>
<tr>
<td><code>bool SimObject::locked</code></td>
</tr>
<tr>
<td><code>string SimObject::name</code></td>
</tr>
<tr>
<td><code>SimObject SimObject::parentGroup</code></td>
</tr>
<tr>
<td><code>pid SimObject::persistentId</code></td>
</tr>
<tr>
<td><code>string SimObject::superClass</code></td>
</tr>
</tbody>
</table>
SimpleMessageEvent Class Reference

[Networking]

A very simple example of a network event. More...

List of all members.
Static Public Member Functions

static void msg (NetConnection con, string message)
Send a SimpleMessageEvent message to the specified connection.
Detailed Description

A very simple example of a network event.

This object exists purely for instructional purposes. It is primarily geared toward developers that wish to understand the inner-working of Torque 3D’s networking system. This is not intended for actual game development.

See also:

NetEvent for the inner workings of network events
Member Function Documentation

```c++
static void SimpleMessageEvent::msg(NetConnection con, string message)
```

Send a `SimpleMessageEvent` message to the specified connection.

The far end that receives the message will print the message out to the console.

**Parameters:**

- `con` The unique ID of the connection to transmit to
- `message` The string containing the message to transmit

**Example:**

```c++
// Send a message to the other end of the NetConnection
SimpleMessageEvent::msg(%conn, "A message from me!");

// The far end will see the following in the console
// (Note: The number may be something other than 1796)
// RMSG 1796  A message from me!
```

Copyright © GarageGames, LLC. All Rights Reserved.
SimpleNetObject Class Reference
[Networking]

A very simple example of a class derived from NetObject. More...

Inheritance diagram for SimpleNetObject:

![Inheritance diagram for SimpleNetObject]

List of all members.
## Public Member Functions

```cpp
void setMessage (string msg)
Sets the internal message variable.
```
Detailed Description

A very simple example of a class derived from NetObject.

This object exists purely for instructional purposes. It is primarily geared toward developers that wish to understand the inner-working of Torque 3D's networking system. This is not intended for actual game development.

Example:

```csharp
// On the server, create a new SimpleNetObject object so it will be immediately ghosted.
$s = new SimpleNetObject();

// All connected clients will see the following in their console:
// Got message: Hello World!
```

See also: NetObject for a full breakdown of this example object.
Member Function Documentation

void SimpleNetObject::setMessage(string msg)

Sets the internal message variable.

SimpleNetObject is set up to automatically transmit this new message to all connected clients. It will appear in the clients' console.

Parameters:

msg The new message to send

Example:

    // On the server, create a new SimpleNetObj
    // object so it will be immediately ghosted
    $s = new SimpleNetObject();

    // All connected clients will see the followin
    // Got message: Hello World!

    // Now again on the server, change the message
    // be sent to all connected clients.
    $s.setMessage("A new message from me!");

    // All connected clients will now see in their conso
    // Go message: A new message from me!
SimSet Class Reference
[Console, Scripting]

A collection of SimObjects. More...

Inheritance diagram for SimSet:
List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bool acceptsAsChild (SimObject obj)</code></td>
<td>Test whether the given object may be added to the set.</td>
</tr>
<tr>
<td><code>void add (SimObject objects...)</code></td>
<td>Add the given objects to the set.</td>
</tr>
<tr>
<td><code>void bringToFront (SimObject obj)</code></td>
<td>Make the given object the first object in the set.</td>
</tr>
<tr>
<td><code>void callOnChildren (string method, string args...)</code></td>
<td>Call a method on all objects contained in the set.</td>
</tr>
<tr>
<td><code>void callOnChildrenNoRecurse (string method, string args...)</code></td>
<td>Call a method on all objects contained in the set.</td>
</tr>
<tr>
<td><code>void clear ()</code></td>
<td>Remove all objects from the set.</td>
</tr>
<tr>
<td><code>void deleteAllObjects ()</code></td>
<td>Delete all objects in the set.</td>
</tr>
<tr>
<td><code>SimObject findObjectByInternalName (string internalName, bool searchChildren=false)</code></td>
<td>Find an object in the set by its internal name.</td>
</tr>
<tr>
<td><code>int getCount ()</code></td>
<td>Get the number of objects contained in the set.</td>
</tr>
<tr>
<td><code>int getFullCount ()</code></td>
<td>Get the number of direct and indirect child objects contained in the set.</td>
</tr>
<tr>
<td><code>SimObject getObject (int index)</code></td>
<td>Get the object at the given index.</td>
</tr>
<tr>
<td><code>int getObjectIndex (SimObject obj)</code></td>
<td>Return the index of the given object in this set.</td>
</tr>
<tr>
<td><code>SimObject getRandom ()</code></td>
<td>Return a random object from the set.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>bool isMember (SimObject obj)</code></td>
<td>Test whether the given object belongs to the set.</td>
</tr>
<tr>
<td><code>void listObjects ()</code></td>
<td>Dump a list of all objects contained in the set to the console.</td>
</tr>
<tr>
<td><code>void pushToBack (SimObject obj)</code></td>
<td>Make the given object the last object in the set.</td>
</tr>
<tr>
<td><code>void remove (SimObject objects...)</code></td>
<td>Remove the given objects from the set.</td>
</tr>
<tr>
<td><code>void reorderChild (SimObject child1, SimObject child2)</code></td>
<td>Make sure child1 is ordered right before child2 in the set.</td>
</tr>
<tr>
<td><code>void sort (string callbackFunction)</code></td>
<td>Sort the objects in the set using the given comparison function.</td>
</tr>
</tbody>
</table>

**Callbacks**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void onObjectAdded (SimObject object)</code></td>
<td>Called when an object is added to the set.</td>
</tr>
<tr>
<td><code>void onObjectRemoved (SimObject object)</code></td>
<td>Called when an object is removed from the set.</td>
</tr>
</tbody>
</table>
Detailed Description

A collection of SimObjects.

It is often necessary to keep track of an arbitrary set of SimObjects. For instance, Torque's networking code needs to not only keep track of the set of objects which need to be ghosted, but also the set of objects which must *always* be ghosted. It does this by working with two sets. The first of these is the RootGroup (which is actually a SimGroup) and the second is the GhostAlwaysSet, which contains objects which must always be ghosted to the client.

Some general notes on SimSets:

- Membership is not exclusive. A SimObject may be a member of multiple SimSets.
- A SimSet does not destroy subobjects when it is destroyed.
- A SimSet may hold an arbitrary number of objects.
Member Function Documentation

bool SimSet::acceptsAsChild(SimObject obj)

Test whether the given object may be added to the set.

**Parameters:**

  * obj  The object to test for potential membership.

**Returns:**

True if the object may be added to the set, false otherwise.

void SimSet::add(SimObject objects...)

Add the given objects to the set.

**Parameters:**

  * objects  The objects to add to the set.

void SimSet::bringToFront(SimObject obj)

Make the given object the first object in the set.

**Parameters:**

  * obj  The object to bring to the frontmost position. Must be contained in the set.

void SimSet::callOnChildren(string method, string args...)

Call a method on all objects contained in the set.
Parameters:

- **method** The name of the method to call.
- **args** The arguments to the method.

**Note:**
This method recurses into all SimSets that are children to the set.

**See also:**
- callOnChildrenNoRecurse

```cpp
void SimSet::callOnChildrenNoRecurse(string method,
                                        string args...)
```

Call a method on all objects contained in the set.

**Parameters:**
- **method** The name of the method to call.
- **args** The arguments to the method.

**Note:**
This method does not recurse into child SimSets.

**See also:**
- callOnChildren

```cpp
void SimSet::clear() 
```

Remove all objects from the set.

Reimplemented in GuiPopUpMenuCtrlEx.
void SimSet::deleteAllObjects()

Delete all objects in the set.

SimObject SimSet::findObjectByInternalName(string internalName,
                                          bool searchChildren = false)

Find an object in the set by its internal name.

Parameters:
  - `internalName`  The internal name of the object to look for.
  - `searchChildren` If true, SimSets contained in the set will be recursively searched for the object.

Returns:
  The object with the given internal name or 0 if no match was found.

int SimSet::getCount()

Get the number of objects contained in the set.

Returns:
  The number of objects contained in the set.

int SimSet::getFullCount()

Get the number of direct and indirect child objects contained in the set.

Returns:
  The number of objects contained in the set as well as in other
sets contained directly or indirectly in the set.

**SimObject SimSet::getObject(int index )**

Get the object at the given index.

**Parameters:**

*index* The object index.

**Returns:**
The object at the given index or -1 if index is out of range.

**int SimSet::getObjectIndex(SimObject obj )**

Return the index of the given object in this set.

**Parameters:**

*obj* The object for which to return the index. Must be contained in the set.

**Returns:**
The index of the object or -1 if the object is not contained in the set.

**SimObject SimSet::getRandom( )**

Return a random object from the set.

**Returns:**
A randomly selected object from the set or -1 if the set is empty.
bool SimSet::isMember(SimObject obj)

Test whether the given object belongs to the set.

**Parameters:**

- `obj` The object.

**Returns:**

True if the object is contained in the set; false otherwise.

void SimSet::listObjects()

Dump a list of all objects contained in the set to the console.

void SimSet::onObjectAdded(SimObject object)

Called when an object is added to the set.

**Parameters:**

- `object` The object that was added.

void SimSet::onObjectRemoved(SimObject object)

Called when an object is removed from the set.

**Parameters:**

- `object` The object that was removed.

void SimSet::pushToBack(SimObject obj)

Make the given object the last object in the set.
### Parameters:

- **obj** The object to bring to the last position. Must be contained in the set.

#### void SimSet::remove(SimObject objects...)

Remove the given objects from the set.

**Parameters:**

- **objects** The objects to remove from the set.

#### void SimSet::reorderChild(SimObject child1, SimObject child2)

Make sure child1 is ordered right before child2 in the set.

**Parameters:**

- **child1** The first child. The object must already be contained in the set.
- **child2** The second child. The object must already be contained in the set.

#### void SimSet::sort(string callbackFunction)

Sort the objects in the set using the given comparison function.

**Parameters:**

- **callbackFunction** Name of a function that takes two object arguments A and B and returns -1 if A is less, 1 if B is less, and 0 if both are equal.
SimXMLDocument Class Reference

[File I/O]

File I/O object used for creating, reading, and writing XML documents. More...

Inheritance diagram for SimXMLDocument:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void addComment</td>
<td>Add the given comment as a child of the document.</td>
</tr>
<tr>
<td>void addData</td>
<td>Add the given text as a child of current Element.</td>
</tr>
<tr>
<td>void addHeader</td>
<td>Add a XML header to a document.</td>
</tr>
<tr>
<td>void addNewElement</td>
<td>Create a new element with the given name as child of current Element's parent and push it onto the Element stack making it the current one.</td>
</tr>
<tr>
<td>void addText</td>
<td>Add the given text as a child of current Element.</td>
</tr>
<tr>
<td>string attribute</td>
<td>Get a string attribute from the current Element on the stack.</td>
</tr>
<tr>
<td>bool attributeExists</td>
<td>Tests if the requested attribute exists.</td>
</tr>
<tr>
<td>float attributeF32</td>
<td>Get float attribute from the current Element on the stack.</td>
</tr>
<tr>
<td>int attributeS32</td>
<td>Get int attribute from the current Element on the stack.</td>
</tr>
<tr>
<td>void clear</td>
<td>Set this document to its default state.</td>
</tr>
<tr>
<td>void clearError</td>
<td>Clear the last error description.</td>
</tr>
<tr>
<td>string elementValue</td>
<td>Get the Element's value if it exists.</td>
</tr>
<tr>
<td>string firstAttribute</td>
<td>Obtain the name of the current Element's first attribute.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>getData()</code></td>
<td>Gets the text from the current Element.</td>
</tr>
<tr>
<td><code>getErrorDesc()</code></td>
<td>Get last error description.</td>
</tr>
<tr>
<td><code>getText()</code></td>
<td>Gets the text from the current Element.</td>
</tr>
<tr>
<td><code>lastAttribute()</code></td>
<td>Obtain the name of the current Element's last attribute.</td>
</tr>
<tr>
<td><code>loadFile(fileName)</code></td>
<td>Load in given filename and prepare it for use.</td>
</tr>
<tr>
<td><code>nextAttribute()</code></td>
<td>Get the name of the next attribute for the current Element after a call to <code>firstAttribute()</code>.</td>
</tr>
<tr>
<td><code>nextSiblingElement(name)</code></td>
<td>Put the next sibling Element with the given name on the stack, making it the current one.</td>
</tr>
<tr>
<td><code>parse(xmlString)</code></td>
<td>Create a document from a XML string.</td>
</tr>
<tr>
<td><code>popElement()</code></td>
<td>Pop the last Element off the stack.</td>
</tr>
<tr>
<td><code>prevAttribute()</code></td>
<td>Get the name of the previous attribute for the current Element after a call to <code>lastAttribute()</code>.</td>
</tr>
<tr>
<td><code>pushChildElement(index)</code></td>
<td>Push the child Element at the given index onto the stack, making it the current one.</td>
</tr>
<tr>
<td><code>pushFirstChildElement(name)</code></td>
<td>Push the first child Element with the given name onto the stack, making it the current Element.</td>
</tr>
<tr>
<td><code>pushNewElement(name)</code></td>
<td>Create a new element with the given name as child of current Element and push it onto the Element stack making</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>readComment (int index)</code></td>
<td>Gives the comment at the specified index, if any.</td>
</tr>
<tr>
<td><code>removeText ()</code></td>
<td>Remove any text on the current Element.</td>
</tr>
<tr>
<td><code>reset ()</code></td>
<td>Set this document to its default state.</td>
</tr>
<tr>
<td><code>saveFile (string fileName)</code></td>
<td>Save document to the given file name.</td>
</tr>
<tr>
<td><code>setAttribute (string attributeName, string value)</code></td>
<td>Set the attribute of the current Element on the stack to the given value.</td>
</tr>
<tr>
<td><code>setObjectAttributes (string objectID)</code></td>
<td>Add the given SimObject's fields as attributes of the current Element on the stack.</td>
</tr>
</tbody>
</table>
Detailed Description

File I/O object used for creating, reading, and writing XML documents.

A SimXMLDocument is a container of various XML nodes. The Document level may contain a header (sometimes called a declaration), comments and child Elements. Elements may contain attributes, data (or text) and child Elements.

You build new Elements using addNewElement(). This makes the new Element the current one you're working with. You then use setAttribute() to add attributes to the Element. You use addData() or addText() to write to the text area of an Element.

Example:

```xml
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<DataTables>
  <table tableName="2DShapes">
    <rec id="1">Triangle</rec>
    <rec id="2">Square</rec>
    <rec id="3">Circle</rec>
  </table>
  <table tableName="3DShapes">
    <rec id="1">Pyramid</rec>
    <rec id="2">Cube</rec>
    <rec id="3">Sphere</rec>
  </table>
</DataTables>
```

```javascript
function readXmlExample(%filename) {
  // Using SimXMLDocument by itself
}
```
```javascript
%xml = new SimXMLDocument() {}; %xml.loadFile(%filename);

%xml.pushChildElement("DataTables");
%xml.pushFirstChildElement("table");
while (true)
{
    echo("TABLE:" SPC %xml.attribute("tableName");
    %xml.pushFirstChildElement("rec");
    while (true)
    {
        %id = %xml.attribute("id");
        %desc = %xml.getData();
        echo(" Shape" SPC %id SPC %desc);
        if (!%xml.nextSiblingElement("rec")
    }
    %xml.popElement();
    if (!%xml.nextSiblingElement("table")
}

// Thanks to Scott Peal for this example
// Using FileObject in conjunction with SimXML
// This example uses an XML file with a format of:
// <Models>
//    <Model category="" name="" path="" />
// </Models>
function getModelsInCategory()
{
    %file = "./Catalog.xml";
    %fo = new FileObject();
    %text = "";
```
if(%fo.openForRead(%file))
{
    while(!%fo.isEOF())
    {
        %text = %text @ %fo.readLine();
        if (!%fo.isEOF()) %text = %text @ "\n"
    }
}
else
{
    echo("Unable to locate the file: " @ ':
    
%fo.delete();

%xml = new SimXMLDocument(){};
%xml.parse(%text);
// "Get" inside of the root element, "Model"
%xml.pushChildElement(0);

// "Get" into the first child element
if (%xml.pushFirstChildElement("Model"))
{
    while (true)
    {
        //
        // Here, i read the element's attributes.
        // You might want to save these values into an array
        // if you have a different XML structure.

        %catagory = %xml.attribute("catagory";
        %name = %xml.attribute("name";
        %path = %xml.attribute("path";
// now, read the next "Model"
if (!%xml.nextSiblingElement("Model

Note:

SimXMLDocument is a wrapper around TinyXml, a standard XML library. If you're familiar with its concepts, you'll find they also apply here.

See also:

FileObject
Member Function Documentation

```cpp
void SimXMLDocument::addComment(string comment)
```

Add the given comment as a child of the document.

**Parameters:**

- `comment` String containing the comment.

**Example:**

```cpp
// Create a new XML document with a header
SimXMLDocument *x = new SimXMLDocument();
x->addHeader();
x->addComment("This is a test comment");
x->addElement("NewElement");
x->saveFile("test.xml");

// Produces the following file:
// <?xml version="1.0" encoding="utf-8" standalone="yes"
// <!--This is a test comment-->
// <NewElement />
```

See also:

- `readComment()`

```cpp
void SimXMLDocument::addData(string text)
```

Add the given text as a child of current Element.

Use `getData()` to retrieve any text from the current Element.

`addData()` and `addText()` may be used interchangeably. As there is no difference between data and text, you may also use...
removeText() to clear any data from the current Element.

**Parameters:**

*text* String containing the text.

**Example:**

```cpp
// Create a new XML document with a header
// with some added data.
%x = new SimXMLDocument();
%x.addHeader();
%x.addNewElement("NewElement");
%x.addData("Some text");
%x.saveFile("test.xml");

// Produces the following file:
// <?xml version="1.0" encoding="utf-8" standalone="yes"?>
// <NewElement>Some text</NewElement>
```

See also:

- `getData()`
- `addText()`
- `getText()`
- `removeText()`

```cpp
void SimXMLDocument::addHeader( )
```

Add a XML header to a document.

Sometimes called a declaration, you typically add a standard header to the document before adding any elements. **SimXMLDocument** always produces the following header:

```xml
<?xml version="1.0" encoding="utf-8" standalone="yes" ?>
```

**Example:**
Create a new XML document with just a header and single element.

```cpp
SimXMLDocument %x = new SimXMLDocument();
%x.addHeader();
%x.addNewElement("NewElement");
%x.saveFile("test.xml");
```

// Produces the following file:
// <?xml version="1.0" encoding="utf-8" standalone="yes" ?>
// <NewElement />

### void SimXMLDocument::addNewElement(string name)

Create a new element with the given name as child of current Element's parent and push it onto the Element stack making it the current one.

**Note:**

This differs from `pushNewElement()` in that it adds the new Element to the current Element's parent (or document if there is no parent Element). This makes the new Element a sibling of the current one.

**Parameters:**

- `name` XML tag for the new Element.

**See also:**

- `pushNewElement()`

### void SimXMLDocument::addText(string text)

Add the given text as a child of current Element.

Use `getText()` to retrieve any text from the current Element and
removeText() to clear any text.

addText() and addData() may be used interchangeably.

**Parameters:**

*text*  String containing the text.

**Example:**

```cpp
// Create a new XML document with a header with some added text.
%x = new SimXMLDocument();
%x.addHeader();
%x.addNewElement("NewElement");
%x.addText("Some text");
%x.saveFile("test.xml");

// Produces the following file:
// <?xml version="1.0" encoding="utf-8" standalone="yes">
// <NewElement>Some text</NewElement>
```

**See also:**

getText()
removeText()
addData()
getData()

```cpp
text SimXMLDocument::attribute(string attributeName )
```

Get a string attribute from the current Element on the stack.

**Parameters:**

*attributeName*  Name of attribute to retrieve.

**Returns:**
The attribute string if found. Otherwise returns an empty string.

```cpp
bool SimXMLDocument::attributeExists(string attributeName)
```

Tests if the requested attribute exists.

**Parameters:**
- `attributeName` Name of attribute being queried for.

**Returns:**
- True if the attribute exists.

```cpp
float SimXMLDocument::attributeF32(string attributeName)
```

Get float attribute from the current Element on the stack.

**Parameters:**
- `attributeName` Name of attribute to retrieve.

**Returns:**
- The value of the given attribute in the form of a float.

```cpp
int SimXMLDocument::attributeS32(string attributeName)
```

Get int attribute from the current Element on the stack.

**Parameters:**
- `attributeName` Name of attribute to retrieve.

**Returns:**
- The value of the given attribute in the form of an integer.
void SimXMLDocument::clear()

Set this document to its default state.
Clears all Elements from the documents. Equivalent to using reset()

See also:
reset()

void SimXMLDocument::clearError()

Clear the last error description.

string SimXMLDocument::elementValue()

Get the Element's value if it exists.
Usually returns the text from the Element.

Returns:
The value from the Element, or an empty string if none is found.

string SimXMLDocument::firstAttribute()

Obtain the name of the current Element's first attribute.

Returns:
String containing the first attribute's name, or an empty string if none is found.

See also:
string SimXMLDocument::getData() 

Gets the text from the current Element.

Use addData() to add text to the current Element.

getData() and getText() may be used interchangeably. As there is no difference between data and text, you may also use removeText() to clear any data from the current Element.

Returns:
String containing the text in the current Element.

Example:

// Using the following test.xml file as an example:
// <?xml version="1.0" encoding="utf-8" standalone="yes"?>
// <NewElement>Some data</NewElement>

// Load in the file
%x = new SimXMLDocument();
%x.loadFile("test.xml");

// Make the first Element the current one
%x.pushFirstChildElement("NewElement");

// Store the current Element's data ('Some data') into 'result'
%result = %x.getData();
echo( %result );
See also:
addData()
addText()
getText()
removeText()

string SimXMLDocument::getErrorDesc( )

Get last error description.

**Returns:**
A string of the last error message.

string SimXMLDocument::getText( )

Gets the text from the current Element.

Use addText() to add text to the current Element and removeText() to clear any text.

**Returns:**
String containing the text in the current Element.

**Example:**

```c++
// Using the following test.xml file as an example:
// <?xml version="1.0" encoding="utf-8" standalone="yes"?>
// <NewElement>Some text</NewElement>

// Load in the file
%x = new SimXMLDocument();
%x.loadFile("test.xml");
```
// Make the first Element the current one
%x.pushFirstChildElement("NewElement");

// Store the current Element's text ('Some
// into 'result'
%result = %x.getText();
echo( %result );

See also:
  addText()
  removeText()
  addData()
  getData()

string SimXMLDocument::lastAttribute( )

Obtain the name of the current Element's last attribute.

Returns:
  String containing the last attribute's name, or an empty string
  if none is found.

See also:
  prevAttribute()
  firstAttribute()
  lastAttribute()

bool SimXMLDocument::loadFile(string fileName )

Load in given filename and prepare it for use.

Note:
  Clears the current document's contents.
Parameters:

fileName  Name and path of XML document

Returns:

True if the file was loaded successfully.

string SimXMLDocument::nextAttribute()  

Get the name of the next attribute for the current Element after a call to firstAttribute().

Returns:

String containing the next attribute's name, or an empty string if none is found.

See also:

firstAttribute()
lastAttribute()
prevAttribute()

bool SimXMLDocument::nextSiblingElement(string name )  

Put the next sibling Element with the given name on the stack, making it the current one.

Parameters:

name  String containing name of the next sibling.

Returns:

True if the Element was found and made the current one.

void SimXMLDocument::parse(string xmlString )
Create a document from a XML string.

**Note:**
Clears the current document's contents.

**Parameters:**
- *xmlString* Valid XML to parse and store as a document.

```cpp
void SimXMLDocument::popElement()  
```

Pop the last Element off the stack.

```cpp
string SimXMLDocument::prevAttribute()  
```

Get the name of the previous attribute for the current Element after a call to `lastAttribute()`.

**Returns:**
String containing the previous attribute's name, or an empty string if none is found.

**See also:**
- `lastAttribute()`
- `firstAttribute()`
- `nextAttribute()`

```cpp
bool SimXMLDocument::pushChildElement(int index)  
```

Push the child Element at the given index onto the stack, making it the current one.

**Parameters:**
Push the first child Element with the given name onto the stack, making it the current Element.

**Parameters:**

- *name*  String containing name of the child Element.

**Returns:**

True if the Element was found and made the current one.

**Example:**

```plaintext
// Using the following test.xml file as an example:
<?xml version="1.0" encoding="utf-8" standalone="yes" ?>
<NewElement>Some text</NewElement>

// Load in the file
%x = new SimXMLDocument();
%x.loadFile("test.xml");

// Make the first Element the current one
%x.pushFirstChildElement("NewElement");

// Store the current Element's text ('Some text' in this example) into 'result'
%result = %x.getText();
echo( %result );
```
void SimXMLDocument::pushNewElement(string name)

Create a new element with the given name as child of current Element and push it onto the Element stack making it the current one.

**Note:**
This differs from `addNewElement()` in that it adds the new Element as a child of the current Element (or a child of the document if no Element exists).

**Parameters:**
- *name*  XML tag for the new Element.

**See also:**
- `addNewElement()`

string SimXMLDocument::readComment(int index)

Gives the comment at the specified index, if any.

Unlike `addComment()` that only works at the document level, `readComment()` may read comments from the document or any child Element. The current Element (or document if no Elements have been pushed to the stack) is the parent for any comments, and the provided index is the number of comments in to read back.

**Parameters:**
- *index*  Comment index number to query from the current Element stack

**Returns:**
- String containing the comment, or an empty string if no comment is found.
void SimXMLDocument::removeText()

Remove any text on the current Element.

Use `getText()` to retrieve any text from the current Element and `addText()` to add text to the current Element. As `getData()` and `addData()` are equivalent to `getText()` and `addText()`, `removeText()` will also remove any data from the current Element.

See also:
addText()
gText()
addData()
getData()

void SimXMLDocument::reset()

Set this document to its default state.

Clears all Elements from the documents. Equivalent to using `clear()`

See also:
clear()

bool SimXMLDocument::saveFile(string fileName)

Save document to the given file name.

Parameters:

`fileName` Path and name of XML file to save to.
Returns:
True if the file was successfully saved.

```cpp
void SimXMLDocument::setAttribute(string attributeName, string value)
```

Set the attribute of the current Element on the stack to the given value.

**Parameters:**
- `attributeName` Name of attribute being changed
- `value` New value to assign to the attribute

```cpp
void SimXMLDocument::setObjectAttributes(string objectId)
```

Add the given SimObject's fields as attributes of the current Element on the stack.

**Parameters:**
- `objectId` ID of SimObject being copied.
SkyBox Class Reference
[Atmosphere]

Represents the sky with an artist-created cubemap. More...

Inheritance diagram for SkyBox:

```
[legend]
```

List of all members.
### Public Attributes

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>postApply</td>
<td></td>
</tr>
</tbody>
</table>

### Sky Box

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>drawBottom</td>
<td>If false the bottom of the skybox is not rendered.</td>
</tr>
<tr>
<td>float</td>
<td>fogBandHeight</td>
<td>The height (0-1) of the fog band from the horizon to the top of the SkyBox.</td>
</tr>
<tr>
<td>string</td>
<td>Material</td>
<td>The name of a cubemap material for the sky box.</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Represents the sky with an artist-created cubemap.

*SkyBox* is not a directional light and should be used in conjunction with a *Sun* object.
## Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>SkyBox::drawBottom</td>
<td>If false the bottom of the skybox is not rendered.</td>
</tr>
<tr>
<td>float</td>
<td>SkyBox::fogBandHeight</td>
<td>The height (0-1) of the fog band from the horizon to the top of the SkyBox.</td>
</tr>
<tr>
<td>string</td>
<td>SkyBox::Material</td>
<td>The name of a cubemap material for the sky box.</td>
</tr>
<tr>
<td>void</td>
<td>SkyBox::postApply</td>
<td></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
SpawnSphere Class Reference
[Game Objects, Miscellaneous]

This class is used for creating any type of game object, assigning it a class, datablock, and other properties when it is spawned. More...

Inheritance diagram for SpawnSphere:

List of all members.
Public Member Functions

bool spawnObject (string additionalProps)
   Dynamically create a new game object with a specified class, datablock, and optional properties.

Callbacks

void onAdd (int objectId)
   Called when the SpawnSphere is being created.
Public Attributes

Spawn

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>autoSpawn</td>
<td>Flag to spawn object as soon as SpawnSphere is created, true to enable or</td>
</tr>
<tr>
<td></td>
<td></td>
<td>false to disable.</td>
</tr>
<tr>
<td>string</td>
<td>spawnClass</td>
<td>Object class to create (eg. Player, AIPlayer, Debris etc).</td>
</tr>
<tr>
<td>string</td>
<td>spawnDatablock</td>
<td>Predefined datablock assigned to the object when created.</td>
</tr>
<tr>
<td>string</td>
<td>spawnProperties</td>
<td>String containing semicolon (;) delimited properties to set when the object</td>
</tr>
<tr>
<td></td>
<td></td>
<td>is created.</td>
</tr>
<tr>
<td>string</td>
<td>spawnScript</td>
<td>Command to execute immediately after spawning an object. New object id is</td>
</tr>
<tr>
<td></td>
<td></td>
<td>stored in $SpawnObject. Max 255 characters.</td>
</tr>
<tr>
<td>bool</td>
<td>spawnTransform</td>
<td>Flag to set the spawned object's transform to the SpawnSphere's transform.</td>
</tr>
</tbody>
</table>

Weight

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>indoorWeight</td>
<td>Deprecated.</td>
</tr>
<tr>
<td>float</td>
<td>outdoorWeight</td>
<td>Deprecated.</td>
</tr>
<tr>
<td>float</td>
<td>sphereWeight</td>
<td>Deprecated.</td>
</tr>
</tbody>
</table>

Dimensions
<table>
<thead>
<tr>
<th>float</th>
<th>radius</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Deprecated.</td>
</tr>
</tbody>
</table>
# Static Public Attributes

<table>
<thead>
<tr>
<th>static bool</th>
<th>isRenderable</th>
<th>Disables rendering of all instances of this type.</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>

---
Detailed Description

This class is used for creating any type of game object, assigning it a class, datablock, and other properties when it is spawned.

Torque 3D uses a simple spawn system, which can be easily modified to spawn any kind of object (of any class). Each new level already contains at least one SpawnSphere, which is represented by a green octahedron in stock Torque 3D. The spawnClass field determines the object type, such as Player, AIPlayer, etc. The spawnDataBlock field applies the pre-defined datablock to each spawned object instance. The really powerful feature of this class is provided by the spawnScript field which allows you to define a simple script (multiple lines) that will be executed once the object has been spawned.

Example:

```cpp
// Define an SpawnSphere that essentially
// $SpawnObject = new Player()
//{
//  dataBlock = "DefaultPlayerData";
//  name = "Bob";
//  lifeTotal = 3;
//};
//echo("Spawned a Player: " @ $SpawnObject)

new SpawnSphere(DefaultSpawnSphere)
{
  spawnClass = "Player";
  spawnDatablock = "DefaultPlayerData";
  spawnScript = "echo("Spawned a Player: "$SpawnObject);
  spawnProperties = "name = "Bob";lifeTotal = 3;"
  autoSpawn = "1";
  dataBlock = "SpawnSphereMarker";
  position = "-0.77266 -19.882 17.8153";
```
rotation = "1 0 0 0";
scale = "1 1 1";
canSave = "1";
canSaveDynamicFields = "1";
};

// Because autoSpawn is set to true in the
// of code will execute AFTER the Player o
echo("Object Spawned");
echo("Hello World");

See also:
MissionMarker
MissionMarkerData
Member Function Documentation

```cpp
void SpawnSphere::onAdd(int objectId)
```

Called when the `SpawnSphere` is being created.

**Parameters:**

`objectId` The unique SimObjectId generated when `SpawnSphere` is created (%this in script)

```cpp
bool SpawnSphere::spawnObject(string additionalProps)
```

Dynamically create a new game object with a specified class, datablock, and optional properties.

This is called on the actual `SpawnSphere`, not to be confused with the `Sim::spawnObject()` global function.

**Parameters:**

Optional set of semicolon delimited `additionalProps` parameters applied to the spawn object during creation.

**Example:**

```cpp
// Use the SpawnSphere::spawnObject function
// No additional properties assigned
%player = DefaultSpawnSphere.spawnObject();
```
### Member Data Documentation

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Member Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>SpawnSphere::autoSpawn</code></td>
<td>Flag to spawn object as soon as <code>SpawnSphere</code> is created, true to enable or false to disable.</td>
</tr>
<tr>
<td>float</td>
<td><code>SpawnSphere::indoorWeight</code></td>
<td>Deprecated.</td>
</tr>
<tr>
<td>float</td>
<td><code>SpawnSphere::outdoorWeight</code></td>
<td>Deprecated.</td>
</tr>
<tr>
<td>float</td>
<td><code>SpawnSphere::radius</code></td>
<td>Deprecated.</td>
</tr>
<tr>
<td>string</td>
<td><code>SpawnSphere::spawnClass</code></td>
<td>Object class to create (eg. Player, AIPlayer, Debris etc).</td>
</tr>
<tr>
<td>string</td>
<td><code>SpawnSphere::spawnDatablock</code></td>
<td>Predefined datablock assigned to the object when created.</td>
</tr>
<tr>
<td>string</td>
<td><code>SpawnSphere::spawnProperties</code></td>
<td></td>
</tr>
</tbody>
</table>
String containing semicolon (;) delimited properties to set when the object is created.

```cpp
string SpawnSphere::spawnScript
```

Command to execute immediately after spawning an object. New object id is stored in $SpawnObject. Max 255 characters.

```cpp
bool SpawnSphere::spawnTransform
```

Flag to set the spawned object's transform to the SpawnSphere's transform.

```cpp
float SpawnSphere::sphereWeight
```

Deprecated.

Copyright © GarageGames, LLC. All Rights Reserved.
Splash Class Reference
[Special Effects]

Manages the ring used for a Splash effect. More...

Inheritance diagram for Splash:

```
    SimObject
      ↓
    NetObject
      ↓
SceneObject
      ↓
GameBase
      ↓
Splash
```

List of all members.
## Static Public Attributes

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Manages the ring used for a Splash effect.
SplashData Class Reference
[Special Effects]

Acts as the physical point in space in white a Splash is created from.

More...

Inheritance diagram for SplashData:

```
[legend]
```

List of all members.
Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>acceleration</td>
<td>Constant acceleration value to place upon the splash effect.</td>
</tr>
<tr>
<td>ColorF</td>
<td>colors [4]</td>
<td>Color values to set the splash effect, rgba. Up to 4 allowed. Will transition through colors based on values set in the times value. Example: colors[0] = &quot;0.6 1.0 1.0 0.5&quot;.</td>
</tr>
<tr>
<td>int</td>
<td>delayMS</td>
<td>Time to delay, in milliseconds, before actually starting this effect.</td>
</tr>
<tr>
<td>int</td>
<td>delayVariance</td>
<td>Time variance for delayMS.</td>
</tr>
<tr>
<td>float</td>
<td>ejectionAngle</td>
<td>Rotational angle to create a splash ring.</td>
</tr>
<tr>
<td>float</td>
<td>ejectionFreq</td>
<td>Frequency in which to emit splash rings.</td>
</tr>
<tr>
<td>ParticleEmitterData</td>
<td>emitter [3]</td>
<td>List of particle emitters to create at the point of this Splash effect.</td>
</tr>
<tr>
<td>ExplosionData</td>
<td>Explosion</td>
<td>ExplosionData object to create at the creation position of this splash effect.</td>
</tr>
<tr>
<td>float</td>
<td>height</td>
<td>Height for the splash to reach.</td>
</tr>
<tr>
<td>int</td>
<td>lifetimeMS</td>
<td>Lifetime for this effect, in milliseconds.</td>
</tr>
<tr>
<td>int</td>
<td>lifetimeVariance</td>
<td>Time variance for lifetimeMS.</td>
</tr>
<tr>
<td>int</td>
<td>numSegments</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>Number of ejection points in the splash ring.</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>float ringLifetime</td>
<td>Lifetime, in milliseconds, for a splash ring.</td>
<td></td>
</tr>
<tr>
<td><strong>Point3F scale</strong></td>
<td>The scale of this splashing effect, defined as the F32 points X, Y, Z.</td>
<td></td>
</tr>
<tr>
<td>SFXProfile soundProfile</td>
<td>SFXProfile effect to play.</td>
<td></td>
</tr>
<tr>
<td>float startRadius</td>
<td>Starting radius size of a splash ring.</td>
<td></td>
</tr>
<tr>
<td>float texFactor</td>
<td>Factor in which to apply the texture to the splash ring, 0.0f - 1.0f.</td>
<td></td>
</tr>
<tr>
<td>filename texture</td>
<td>Imagemap file to use as the texture for the splash effect.</td>
<td></td>
</tr>
<tr>
<td>float texWrap</td>
<td>Amount to wrap the texture around the splash ring, 0.0f - 1.0f.</td>
<td></td>
</tr>
<tr>
<td>float times [4]</td>
<td>Times to transition through the splash effect. Up to 4 allowed. Values are 0.0 - 1.0, and correspond to the life of the particle where 0 is first created and 1 is end of lifespace.</td>
<td></td>
</tr>
<tr>
<td>float velocity</td>
<td>Velocity for the splash effect to travel.</td>
<td></td>
</tr>
<tr>
<td>float width</td>
<td>Width for the X and Y coordinates to create this effect within.</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

Acts as the physical point in space in white a Splash is created from.
Member Data Documentation

**float SplashData::acceleration**

Constant acceleration value to place upon the splash effect.

**ColorF SplashData::colors[4]**

Color values to set the splash effect, rgba. Up to 4 allowed. Will transition through colors based on values set in the times value. Example: colors[0] = "0.6 1.0 1.0 0.5".

**int SplashData::delayMS**

Time to delay, in milliseconds, before actually starting this effect.

**int SplashData::delayVariance**

Time variance for delayMS.

**float SplashData::ejectionAngle**

Rotational angle to create a splash ring.

**float SplashData::ejectionFreq**

Frequency in which to emit splash rings.

**ParticleEmitterData SplashData::emitter[3]**
List of particle emitters to create at the point of this *Splash* effect.

**ExplosionData SplashData::Explosion**

*ExplosionData* object to create at the creation position of this splash effect.

**float SplashData::height**

Height for the splash to reach.

**int SplashData::lifetimeMS**

Lifetime for this effect, in milliseconds.

**int SplashData::lifetimeVariance**

Time variance for lifetimeMS.

**int SplashData::numSegments**

Number of ejection points in the splash ring.

**float SplashData::ringLifetime**

Lifetime, in milliseconds, for a splash ring.

**Point3F SplashData::scale**
The scale of this splashing effect, defined as the F32 points X, Y, Z.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SFXProfile SplashData::soundProfile</code></td>
<td>Effect to play.</td>
<td></td>
</tr>
<tr>
<td><code>float SplashData::startRadius</code></td>
<td>Starting radius size of a splash ring.</td>
<td></td>
</tr>
<tr>
<td><code>float SplashData::texFactor</code></td>
<td>Factor in which to apply the texture to the splash ring, 0.0f - 1.0f.</td>
<td></td>
</tr>
<tr>
<td><code>filename SplashData::texture[2]</code></td>
<td>Imagemap file to use as the texture for the splash effect.</td>
<td></td>
</tr>
<tr>
<td><code>float SplashData::texWrap</code></td>
<td>Amount to wrap the texture around the splash ring, 0.0f - 1.0f.</td>
<td></td>
</tr>
</tbody>
</table>
| `float SplashData::times[4]` | Times to transition through the splash effect. Up to 4 allowed. Values are 0.0 - 1.0, and correspond to the life of the particle where 0 is first created and 1 is end of lifespac.
float SplashData::velocity

Velocity for the splash effect to travel.

float SplashData::width

Width for the X and Y coordinates to create this effect within.
Lighting object which emits conical light in a direction. More...

Inheritance diagram for SpotLight:

```
SimObject
  ↓
NetObject
  ↓
SceneObject
  ↓
LightBase
  ↓
SpotLight
```

List of all members.
Public Attributes

Light

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>innerAngle</td>
</tr>
<tr>
<td>float</td>
<td>outerAngle</td>
</tr>
<tr>
<td>float</td>
<td>range</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Lighting object which emits conical light in a direction.

SpotLight is one of the two types of lighting objects that can be added to a Torque 3D level, the other being PointLight. Unlike directional or point lights, the SpotLights emits lighting in a specific direction within a cone. The distance of the cone is controlled by the SpotLight::range variable.

Example:

```cpp
// Declaration of a point light in script,
new SpotLight(SampleSpotLight)
{
    range = "10";
    innerAngle = "40";
    outerAngle = "45";
    isEnabled = "1";
    color = "1 1 1 1";
    brightness = "1";
    castShadows = "0";
    priority = "1";
    animate = "1";
    animationPeriod = "1";
    animationPhase = "1";
    flareType = "LightFlareExample0";
    flareScale = "1";
    attenuationRatio = "0 1 1";
    shadowType = "Spot";
    texSize = "512";
    overDarkFactor = "2000 1000 500 100";
    shadowDistance = "400";
    shadowSoftness = "0.15";
    numSplits = "1";
}
```
logWeight = "0.91";
fadeStartDistance = "0";
lastSplitTerrainOnly = "0"
representedInLightmap = "0"
shadowDarkenColor = "0 0 0 -1"
includeLightmappedGeometryInShadow = "0"
position = "-29.4362 -5.86289 5.58602"
rotation = "1 0 0 0";

};

See also:

LightBase
PointLight
Member Data Documentation

float SpotLight::innerAngle

float SpotLight::outerAngle

float SpotLight::range

Copyright © GarageGames, LLC. All Rights Reserved.
StaticShape Class Reference
[Game Objects]

The most basic 3D shape with a datablock available in Torque 3D.
More...

Inheritance diagram for StaticShape:

List of all members.
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>

**Detailed Description**

The most basic 3D shape with a datablock available in Torque 3D.

When it comes to placing 3D objects in the scene, you technically have two options:

1. **TSStatic** objects

2. **ShapeBase** derived objects

Since **ShapeBase** and **ShapeBaseData** are not meant to be instantiated in script, you will use one of its child classes instead. Several game related objects are derived from **ShapeBase: Player, Vehicle, Item**, and so on.

When you need a 3D object with datablock capabilities, you will use an object derived from **ShapeBase**. When you need an object with extremely low overhead, and with no other purpose than to be a 3D object in the scene, you will use **TSStatic**.

The most basic child of **ShapeBase** is **StaticShape**. It does not introduce any of the additional functionality you see in **Player, Item, Vehicle** or the other game play heavy classes. At its core, it is comparable to a **TSStatic**, but with a datbalock. Having a datablock provides a location for common variables as well as having access to various **ShapeBaseData, GameBaseData** and **SimDataBlock** callbacks.

**Example:**

```cpp
// Create a StaticShape using a datablock
datablock StaticShapeData(BasicShapeData)
{
    shapeFile = "art/shapes/items/kit/healthkit.dts"
    testVar = "Simple string, not a stock variable"
};
```
new StaticShape()
{
    dataBlock = "BasicShapeData";
    position = "0.0 0.0 0.0";
    rotation = "1 0 0 0";
    scale = "1 1 1";
};

See also:
StaticShapeData
ShapeBase
TSStatic

Copyright © GarageGames, LLC. All Rights Reserved.
The most basic *ShapeBaseData* derived shape datablock available in Torque 3D. More...

Inheritance diagram for StaticShapeData:

```
        SimObject
         ↑
        ↓
  SimDataBlock
      ↑         ↓
GameBaseData        StaticShapeData
```

List of all members.
<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>dynamicType</td>
<td>An integer value which, if specified, is added to the value returned by getType().</td>
</tr>
<tr>
<td>bool</td>
<td>noIndividualDamage</td>
<td>Deprecated.</td>
</tr>
</tbody>
</table>
Detailed Description

The most basic ShapeBaseData derived shape datablock available in Torque 3D.

When it comes to placing 3D objects in the scene, you effectively have two options:

1. TSStatic objects

2. ShapeBase derived objects

Since ShapeBase and ShapeBaseData are not meant to be instantiated in script, you will use one of its child classes instead. Several game related objects are derived from ShapeBase: Player, Vehicle, Item, and so on.

When you need a 3D object with datablock capabilities, you will use an object derived from ShapeBase. When you need an object with extremely low overhead, and with no other purpose than to be a 3D object in the scene, you will use TSStatic.

The most basic child of ShapeBase is StaticShape. It does not introduce any of the additional functionality you see in Player, Item, Vehicle or the other game play heavy classes. At its core, it is comparable to a TSStatic, but with a datablock. Having a datablock provides a location for common variables as well as having access to various ShapeBaseData, GameBaseData and SimDataBlock callbacks.

Example:

```cpp
// Create a StaticShape using a datablock
datablock StaticShapeData(BasicShapeData) {
    shapeFile = "art/shapes/items/kit/healthkit.dts"
    testVar = "Simple string, not a stock var";
};
```
new StaticShape()
{
    dataBlock = "BasicShapeData";
    position = "0.0 0.0 0.0";
    rotation = "1 0 0 0";
    scale = "1 1 1";
};

See also:

StaticShape
ShapeBaseData
TSStatic
Member Data Documentation

**int StaticShapeData::dynamicType**

An integer value which, if specified, is added to the value returned by `getType()`.

This allows you to extend the type mask for a `StaticShape` that uses this datablock. Type masks are used for container queries, etc.

**bool StaticShapeData::noIndividualDamage**

Deprecated.
StreamObject Class Reference
[File I/O]

Base class for working with streams. More...

Inheritance diagram for StreamObject:

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bool copyFrom (SimObject other)</code></td>
<td>Copy from another StreamObject into this StreamObject.</td>
</tr>
<tr>
<td><code>int getPosition ()</code></td>
<td>Gets the position in the stream.</td>
</tr>
<tr>
<td><code>string getStatus ()</code></td>
<td>Gets a printable string form of the stream's status.</td>
</tr>
<tr>
<td><code>int getStreamSize ()</code></td>
<td>Gets the size of the stream.</td>
</tr>
<tr>
<td><code>bool isEOF ()</code></td>
<td>Tests if the stream has reached the end of the file.</td>
</tr>
<tr>
<td><code>bool isEOS ()</code></td>
<td>Tests if the stream has reached the end of the file.</td>
</tr>
<tr>
<td><code>string readLine ()</code></td>
<td>Read a line from the stream.</td>
</tr>
<tr>
<td><code>String readLongString (int maxLength)</code></td>
<td>Read in a string up to the given maximum number of characters.</td>
</tr>
<tr>
<td><code>String readString ()</code></td>
<td>Read a string up to a maximum of 256 characters.</td>
</tr>
<tr>
<td><code>String readSTString (bool caseSensitive=false)</code></td>
<td>Read in a string and place it on the string table.</td>
</tr>
<tr>
<td><code>bool setPosition (int newPosition)</code></td>
<td>Gets the position in the stream.</td>
</tr>
<tr>
<td><code>void writeLine (string line)</code></td>
<td>Write a line to the stream, if it was opened for writing.</td>
</tr>
<tr>
<td><code>void writeLongString (int maxLength, string string)</code></td>
<td>Write out a string up to the maximum number of characters.</td>
</tr>
<tr>
<td><code>void writeString (string string, int maxLength=256)</code></td>
<td></td>
</tr>
</tbody>
</table>
Write out a string with a default maximum length of 256 characters.
**Detailed Description**

Base class for working with streams.

You do not instantiate a `StreamObject` directly. Instead, it is used as part of a `FileStreamObject` and `ZipObject` to support working with uncompressed and compressed files respectively.

**Example:**

```cpp
// You cannot actually declare a StreamObject
// Instead, use the derived class "FileStreamObject"
%fsObject = FileStreamObject();
```

**See also:**

- `FileStreamObject` for the derived class usable in script.
- `ZipObject` where `StreamObject` is used to read and write to files within a zip archive.
Member Function Documentation

```cpp
bool StreamObject::copyFrom(SimObject other)
```

Copy from another `StreamObject` into this `StreamObject`.

**Parameters:**
- `other` The `StreamObject` to copy from.

**Returns:**
- True if the copy was successful.

```cpp
int StreamObject::getPosition()
```

Gets the position in the stream.

The easiest way to visualize this is to think of a cursor in a text file. If you have moved the cursor by five characters, the current position is 5. If you move ahead 10 more characters, the position is now 15. For `StreamObject`, when you read in the line the position is increased by the number of characters parsed, the null terminator, and a newline.

**Example:**

```cpp
// Create a file stream object for reading
%fsObject = new FileStreamObject();

// Open a file for reading
// This file contains two lines of text repeating
// Hello World
// Hello World
%fsObject.open("./test.txt", "read");

// Read in the first line
```
%line = %fsObject.readLine();

// Get the position of the stream
%position = %fsObject.getPosition();

// Print the current position
// Should be 13, 10 for the words, 1 for \t
echo(%position);

// Always remember to close a file stream
%fsObject.close();

**Returns:**
Number of bytes which stream has parsed so far, null terminators and newlines are included

**See also:**
setPosition()

---

**string StreamObject::getStatus( )**

Gets a printable string form of the stream's status.

**Example:**

```
// Create a file stream object for reading
%fsObject = new FileStreamObject();

// Open a file for reading
%fsObject.open("./test.txt", "read");

// Get the status and print it
%status = %fsObject.getStatus();
echo(%status);
```
// Always remember to close a file stream
%fsObject.close();

**Returns:**
String containing status constant, one of the following:

OK - Stream is active and no file errors

IOError - Something went wrong during read or writing the stream

EOS - End of Stream reached (mostly for reads)

IllegalCall - An unsupported operation used. Always w/ accompanied by AssertWarn

Closed - Tried to operate on a closed stream (or detached filter)

UnknownError - Catch all for an error of some kind

Invalid - Entire stream is invalid

```cpp
int StreamObject::getStreamSize()
```

Gets the size of the stream.

The size is dependent on the type of stream being used. If it is a file stream, returned value will be the size of the file. If it is a memory stream, it will be the size of the allocated buffer.

**Example:**
```cpp
// Create a file stream object for reading
%fsObject = new FileStreamObject();

// Open a file for reading
// This file contains the following two lines:
```
// HelloWorld
// HelloWorld
%fsObject.open("./test.txt", "read");

// Found out how large the file stream is
// Then print it to the console
// Should be 22
%streamSize = %fsObject.getStreamSize();
echo(%streamSize);

// Always remember to close a file stream
%fsObject.close();

**Returns:**
Size of stream, in bytes

```c++
bool StreamObject::isEOF() 
```

Tests if the stream has reached the end of the file.

This is an alternative name for isEOS. Both functions are interchangeable. This simply exists for those familiar with some C++ file I/O standards.

**Example:**
```c++
// Create a file stream object for reading
%fsObject = new FileStreamObject();

// Open a file for reading
%fsObject.open("./test.txt", "read");

// Keep reading until we reach the end of
while( !%fsObject.isEOF() )
```
{  
    %line = %fsObject.readLine();  
    echo(%line);  
}

// Made it to the end  
echo("Finished reading file");  

// Always remember to close a file stream  
%fsObject.close();

Returns:  
True if the parser has reached the end of the file, false otherwise

See also:  
isEOS()
while(!%fsObject.isEOS()) {
    %line = %fsObject.readLine();
    echo(%line);
}
// Made it to the end
echo("Finished reading file");

// Always remember to close a file stream
%fsObject.close();

**Returns:**

True if the parser has reached the end of the file, false otherwise

**See also:**

isEOF()

**string StreamObject::readLine( )**

Read a line from the stream.

Emphasis on *line*, as in you cannot parse individual characters or chunks of data. There is no limitation as to what kind of data you can read.

**Example:**

```
// Create a file stream object for reading
// This file contains the following two lines:
// HelloWorld
// HelloWorld
%fsObject = new FileStreamObject();
```
%fsObject.open("./test.txt", "read");

// Read in the first line
%line = %fsObject.readLine();

// Print the line we just read
echo(%line);

// Always remember to close a file stream
%fsObject.close();

**Returns:**
String containing the line of data that was just read

**See also:**
writeLine()

---

String StreamObject::readLongString(int maxLength)

Read in a string up to the given maximum number of characters.

**Parameters:**

`maxLength` The maximum number of characters to read in.

**Returns:**
The string that was read from the stream.

**See also:**
writeLongString()

**Note:**
When working with these particular string reading and writing methods, the stream begins with the length of the string followed by the string itself, and does not include a NULL
String StreamObject::readString()

Read a string up to a maximum of 256 characters.

**Returns:**

The string that was read from the stream.

**See also:**

writeString()

**Note:**

When working with these particular string reading and writing methods, the stream begins with the length of the string followed by the string itself, and does not include a NULL terminator.

String StreamObject::readSTString(bool caseSensitive = false)

Read in a string and place it on the string table.

**Parameters:**

If false then case will not be taken into account

*caseSensitive* when attempting to match the read in string with what is already in the string table.

**Returns:**

The string that was read from the stream.

**See also:**

writeString()

**Note:**
When working with these particular string reading and writing
methods, the stream begins with the length of the string
followed by the string itself, and does not include a NULL
terminator.

```cpp
bool StreamObject::setPosition(int newPosition)
```

Gets the position in the stream.

The easiest way to visualize this is to think of a cursor in a text file. If you have moved the cursor by five characters, the current position is 5. If you move ahead 10 more characters, the position is now 15. For `StreamObject`, when you read in the line the position is increased by the number of characters parsed, the null terminator, and a newline. Using `setPosition` allows you to skip to specific points of the file.

Example:

```cpp
// Create a file stream object for reading
%fsObject = new FileStreamObject();

// Open a file for reading
// This file contains the following two lines:
// 11111111111
// Hello World
%fsObject.open("./test.txt", "read");

// Skip ahead by 12, which will bypass the first line
%fsObject.setPosition(12);

// Read in the next line
%line = %fsObject.readLine();

// Print the line just read in, should be "Hello World"
```
echo(%line);

// Always remember to close a file stream
%fsObject.close();

Returns:
Number of bytes which stream has parsed so far, null terminators and newlines are included

See also:
ggetPosition()

void StreamObject::writeLine(string line)

Write a line to the stream, if it was opened for writing.

There is no limit as to what kind of data you can write. Any format and data is allowable, not just text. Be careful of what you write, as whitespace, current values, and literals will be preserved.

Parameters:

  line  The data we are writing out to file.

Example:

  // Create a file stream
  %fsObject = new FileStreamObject();

  // Open the file for writing
  // If it does not exist, it is created. If
  %fsObject.open("./test.txt", "write");

  // Write a line to the file
  %fsObject.writeLine("Hello World");
// Write another line to the file
%fsObject.writeLine("Documentation Rocks!");

// Always remember to close a file stream
%fsObject.close();

See also:
readLine()

void StreamObject::writeLongString(int maxLength,
                                  string string
                                  )

Write out a string up to the maximum number of characters.

Parameters:
  maxLength  The maximum number of characters that will be written.
  string     The string to write out to the stream.

See also:
readLongString()

Note:
When working with these particular string reading and writing methods, the stream begins with the length of the string followed by the string itself, and does not include a NULL terminator.

void StreamObject::writeString(string string,
                                int maxLength = 256
                                )
Write out a string with a default maximum length of 256 characters.

**Parameters:**

- `string`: The string to write out to the stream
- `maxLength`: The maximum string length to write out with a default of 256 characters. This value should not be larger than 256 as it is written to the stream as a single byte.

**See also:**

- `readString()`

**Note:**

When working with these particular string reading and writing methods, the stream begins with the length of the string followed by the string itself, and does not include a NULL terminator.
Sun Class Reference
[Atmosphere]

A global light affecting your entire scene and optionally renders a corona effect. More...

Inheritance diagram for Sun:

List of all members.
### Public Attributes

```c
void animate
animate( F32 duration, F32 startAzimuth, F32 endAzimuth, F32 startElevation, F32 endElevation )
void apply
```

### Lighting

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorF</td>
<td>ambient</td>
<td>Color shading applied to surfaces not in direct contact with light source, such as in the shadows or interiors.</td>
</tr>
<tr>
<td>float</td>
<td>brightness</td>
<td>Adjust the Sun's global contrast/intensity.</td>
</tr>
<tr>
<td>bool</td>
<td>castShadows</td>
<td>Enables/disables shadows cast by objects due to Sun light.</td>
</tr>
<tr>
<td>ColorF</td>
<td>color</td>
<td>Color shading applied to surfaces in direct contact with light source.</td>
</tr>
</tbody>
</table>

### Advanced Lighting

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point3F</td>
<td>attenuationRatio</td>
<td>The proportions of constant, linear, and quadratic attenuation to use for the falloff for point and spot lights.</td>
</tr>
<tr>
<td>filename</td>
<td>cookie</td>
<td>A custom pattern texture which is projected from the light.</td>
</tr>
<tr>
<td>float</td>
<td>fadeStartDistance</td>
<td>Start fading shadows out at this distance. 0 = auto</td>
</tr>
</tbody>
</table>
calculate this distance.

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>lastSplitTerrainOnly</td>
<td>This toggles only terrain being rendered to the last split of a PSSM shadow map.</td>
</tr>
<tr>
<td>float</td>
<td>logWeight</td>
<td>The logarithmic PSSM split distance factor.</td>
</tr>
<tr>
<td>int</td>
<td>numSplits</td>
<td>The logarithmic PSSM split distance factor.</td>
</tr>
<tr>
<td>Point4F</td>
<td>overDarkFactor</td>
<td>The ESM shadow darkening factor.</td>
</tr>
<tr>
<td>float</td>
<td>shadowDistance</td>
<td>The distance from the camera to extend the PSSM shadow.</td>
</tr>
<tr>
<td>float</td>
<td>shadowSoftness</td>
<td></td>
</tr>
<tr>
<td>ShadowType</td>
<td>shadowType</td>
<td>The type of shadow to use on this light.</td>
</tr>
<tr>
<td>int</td>
<td>texSize</td>
<td>The texture size of the shadow map.</td>
</tr>
</tbody>
</table>

**Orbit**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>azimuth</td>
<td>The horizontal angle of the sun measured clockwise from the positive Y world axis.</td>
</tr>
<tr>
<td>float</td>
<td>elevation</td>
<td>The elevation angle of the sun above or below the horizon.</td>
</tr>
</tbody>
</table>

**Corona**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>coronaEnabled</td>
<td>Enable or disable rendering of the corona sprite.</td>
</tr>
<tr>
<td>data type</td>
<td>variable</td>
<td>description</td>
</tr>
<tr>
<td>-----------</td>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>string</td>
<td>coronaMaterial</td>
<td>Texture for the corona sprite.</td>
</tr>
<tr>
<td>float</td>
<td>coronaScale</td>
<td>Controls size the corona sprite renders, specified as a fractional amount of the screen height.</td>
</tr>
<tr>
<td>ColorF</td>
<td>coronaTint</td>
<td>Modulates the corona sprite color (if coronaUseLightColor is false).</td>
</tr>
<tr>
<td>bool</td>
<td>coronaUseLightColor</td>
<td>Modulate the corona sprite color by the color of the light (overrides coronaTint).</td>
</tr>
</tbody>
</table>

**Misc**

<table>
<thead>
<tr>
<th>float</th>
<th>flareScale</th>
<th>Changes the size and intensity of the flare.</th>
</tr>
</thead>
<tbody>
<tr>
<td>LightFlareData</td>
<td>flareType</td>
<td>Datablock for the flare produced by the Sun.</td>
</tr>
</tbody>
</table>

**Advanced Lighting Lightmap**

<table>
<thead>
<tr>
<th>bool</th>
<th>includeLightmappedGeometryInShadow</th>
<th>This light should render lightmapped geometry during its shadow-map update (ignored if 'representedInLightmap' is false).</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>representedInLightmap</td>
<td>This light is represented in lightmaps (static light, default: false).</td>
</tr>
<tr>
<td>ColorF</td>
<td>shadowDarkenColor</td>
<td>The color that should be used to multiply-blend dynamic shadows onto lightmapped geometry (ignored if 'representedInLightmap' is false).</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>static bool isRenderable</th>
<th>Enables rendering of all instances of this type.</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool isSelectable</td>
<td>Enables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
**Detailed Description**

A global light affecting your entire scene and optionally renders a corona effect.

**Sun** is both the directional and ambient light for your entire scene.
## Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorF</td>
<td>Sun::ambient</td>
<td>Color shading applied to surfaces not in direct contact with light source, such as in the shadows or interiors.</td>
</tr>
<tr>
<td>void</td>
<td>Sun::animate</td>
<td>animate( F32 duration, F32 startAzimuth, F32 endAzimuth, F32 startElevation, F32 endElevation )</td>
</tr>
<tr>
<td>void</td>
<td>Sun::apply</td>
<td></td>
</tr>
<tr>
<td>Point3F</td>
<td>Sun::attenuationRatio</td>
<td>The proportions of constant, linear, and quadratic attenuation to use for the falloff for point and spot lights.</td>
</tr>
<tr>
<td>float</td>
<td>Sun::azimuth</td>
<td>The horizontal angle of the sun measured clockwise from the positive Y world axis.</td>
</tr>
<tr>
<td>float</td>
<td>Sun::brightness</td>
<td>Adjust the Sun's global contrast/intensity.</td>
</tr>
<tr>
<td>bool</td>
<td>Sun::castShadows</td>
<td></td>
</tr>
</tbody>
</table>
Enables/disables shadows cast by objects due to Sun light.

**ColorF Sun::color**

Color shading applied to surfaces in direct contact with light source.

**filename Sun::cookie**

A custom pattern texture which is projected from the light.

**bool Sun::coronaEnabled**

Enable or disable rendering of the corona sprite.

**string Sun::coronaMaterial**

Texture for the corona sprite.

**float Sun::coronaScale**

Controls size the corona sprite renders, specified as a fractional amount of the screen height.

**ColorF Sun::coronaTint**

Modulates the corona sprite color (if coronaUseLightColor is false).
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool Sun::coronaUseLightColor</td>
<td>Modulate the corona sprite color by the color of the light (overrides coronaTint).</td>
</tr>
<tr>
<td>float Sun::elevation</td>
<td>The elevation angle of the sun above or below the horizon.</td>
</tr>
<tr>
<td>float Sun::fadeStartDistance</td>
<td>Start fading shadows out at this distance. 0 = auto calculate this distance.</td>
</tr>
<tr>
<td>float Sun::flareScale</td>
<td>Changes the size and intensity of the flare.</td>
</tr>
<tr>
<td>LightFlareData Sun::flareType</td>
<td>Datablock for the flare produced by the Sun.</td>
</tr>
<tr>
<td>bool Sun::includeLightmappedGeometryInShadow</td>
<td>This light should render lightmapped geometry during its shadow-map update (ignored if 'representedInLightmap' is false).</td>
</tr>
<tr>
<td>bool Sun::lastSplitTerrainOnly</td>
<td></td>
</tr>
</tbody>
</table>
This toggles only terrain being rendered to the last split of a PSSM shadow map.

**float** `Sun::logWeight`

The logarithmic PSSM split distance factor.

**int** `Sun::numSplits`

The logarithmic PSSM split distance factor.

**Point4F** `Sun::overDarkFactor`

The ESM shadow darkening factor.

**bool** `Sun::representedInLightmap`

This light is represented in lightmaps (static light, default: false).

**ColorF** `Sun::shadowDarkenColor`

The color that should be used to multiply-blend dynamic shadows onto lightmapped geometry (ignored if ‘representedInLightmap’ is false).

**float** `Sun::shadowDistance`

The distance from the camera to extend the PSSM shadow.
float Sun::shadowSoftness

ShadowType Sun::shadowType

The type of shadow to use on this light.

int Sun::texSize

The texture size of the shadow map.

Copyright © GarageGames, LLC. All Rights Reserved.
TCPObject Class Reference
[Networking]

Allows communications between the game and a server using TCP/IP protocols. More...

Inheritance diagram for TCPObjec:

SimObject

TCPObjec

HTTPObject

[legend]

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>connect</code> (string address)</td>
<td>Connect to the given address.</td>
</tr>
<tr>
<td><code>disconnect</code> ()</td>
<td>Disconnect from whatever this TCPObjec is currently connected to, if anything.</td>
</tr>
<tr>
<td><code>listen</code> (int port)</td>
<td>Start listening on the specified port for connections.</td>
</tr>
<tr>
<td><code>send</code> (string data)</td>
<td>Transmits the data string to the connected computer.</td>
</tr>
</tbody>
</table>

### Callbacks

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>onConnected</code> ()</td>
<td>Called whenever a connection is established with a server.</td>
</tr>
<tr>
<td><code>onConnectFailed</code> ()</td>
<td>Called whenever a connection has failed to be established with a server.</td>
</tr>
<tr>
<td><code>onConnectionRequest</code> (string address, string ID)</td>
<td>Called whenever a connection request is made.</td>
</tr>
<tr>
<td><code>onDisconnect</code> ()</td>
<td>Called whenever the TCPObjec disconnects from whatever it is currently connected to.</td>
</tr>
<tr>
<td><code>onDNSFailed</code> ()</td>
<td>Called whenever the DNS has failed to resolve.</td>
</tr>
<tr>
<td><code>onDNSResolved</code> ()</td>
<td>Called whenever the DNS has been resolved.</td>
</tr>
<tr>
<td><code>onLine</code> (string line)</td>
<td>Called whenever a line of data is sent to this TCPObjec.</td>
</tr>
</tbody>
</table>
Detailed Description

Allows communications between the game and a server using TCP/IP protocols.

To use **TCPObj**ect you set up a connection to a server, send data to the server, and handle each line of the server's response using a callback. Once you are done communicating with the server, you disconnect.

**TCPObj**ect is intended to be used with text based protocols which means you'll need to delineate the server's response with an end-of-line character. i.e. the newline character `\n`. You may optionally include the carriage return character `\r` prior to the newline and **TCPObj**ect will strip it out before sending the line to the callback. If a newline character is not included in the server's output, the received data will not be processed until you disconnect from the server (which flushes the internal buffer).

**TCPObj**ect may also be set up to listen to a specific port, making Torque into a TCP server. When used in this manner, a callback is received when a client connection is made. Following the outside connection, text may be sent and lines are processed in the usual manner.

If you want to work with HTTP you may wish to use **HTTPObj**ect instead as it handles all of the HTTP header setup and parsing.

Example:

```c
// In this example we'll retrieve the new // feed from garagegames.com. As we're us. // raw text response will be received from // the HTTP header.

// Define callbacks for our specific TCPOb // name (RSSFeed) as the namespace.
```
// Handle an issue with resolving the server's name
function RSSFeed::onDNSFailed(%this)
{
    // Store this state
    %this.lastState = "DNSFailed";

    // Handle DNS failure
}

function RSSFeed::onConnectFailed(%this)
{
    // Store this state
    %this.lastState = "ConnectFailed";

    // Handle connection failure
}

function RSSFeed::onDNSResolved(%this)
{
    // Store this state
    %this.lastState = "DNSResolved";
}

function RSSFeed::onConnected(%this)
{
    // Store this state
    %this.lastState = "Connected";
}

function RSSFeed::onDisconnect(%this)
{
// Store this state
%this.lastState = "Disconnected";

} // Handle a line from the server
function RSSFeed::onLine(%this, %line)
{
   // Print the line to the console
   echo( %line );
}

// Create the TCPObject
%rss = new TCPObject(RSSFeed);

// Define a dynamic field to store the last connection state
%rss.lastState = "None";

// Connect to the server
%rss.connect("www.garagegames.com:80");

// Send the RSS feed request to the server
// handled in onLine() callback above
%rss.send("GET /feeds/rss/threads HTTP/1.1

See also:
HTTPObject
Member Function Documentation

void TCPObject::connect(string address )

Connect to the given address.

Parameters:

  address Server address (including port) to connect to.

Example:

```cpp
// Set the address.
%address = "www.garagegames.com:80";

// Inform this TCPObject to connect to the
%thisTCPObj.connect(%address);
```

void TCPObject::disconnect( )

Disconnect from whatever this TCPObject is currently connected to, if anything.

Example:

```cpp
// Inform this TCPObject to disconnect from
%thisTCPObj.disconnect();
```

void TCPObject::listen(int port )

Start listening on the specified port for connections.

This method starts a listener which looks for incoming TCP connections to a port. You must overload the onConnectionRequest callback to create a new TCPObject to
read, write, or reject the new connection.

**Parameters:**

*port*  
Port for this TCPObj ect to start listening for connections on.

**Example:**

```c
// Create a listener on port 8080.  
new TCPObj ect( TCPListener );  
TCPListener.listen( 8080 );

function TCPListener::onConnectionRequest( )  
{  
    // Create a new object to manage the connection.  
    new TCPObj ect( TCPClient, %id );  
}

function TCPClient::onLine( %this, %line );  
{  
    // Print the line of text from client.  
    echo( %line );  
}
```

**void TCPObj ect::onConnected( )**

Called whenever a connection is established with a server.

**void TCPObj ect::onConnectFailed( )**

Called whenever a connection has failed to be established with a server.
void TCPObject::onConnectionRequest(string address, string ID)

Called whenever a connection request is made.

This callback is used when the TCPObject is listening to a port and a client is attempting to connect.

Parameters:

- **address**
  Server address connecting from.
- **ID**
  Connection ID

See also:

listen()

void TCPObject::onDisconnect()

Called whenever the TCPObject disconnects from whatever it is currently connected to.

void TCPObject::onDNSFailed()

Called whenever the DNS has failed to resolve.

void TCPObject::onDNSResolved()

Called whenever the DNS has been resolved.

void TCPObject::onLine(string line)

Called whenever a line of data is sent to this TCPObject.
This callback is called when the received data contains a newline \n character, or the connection has been disconnected and the TCPObj's buffer is flushed.

**Parameters:**

- `line` Data sent from the server.

```cpp
void TCPObj::send(string data )
```

Transmits the data string to the connected computer.

This method is used to send text data to the connected computer regardless if we initiated the connection using `connect()`, or listening to a port using `listen()`.

**Parameters:**

- `data` The data string to send.

**Example:**

```cpp
// Set the command data
%data = "GET " @ $RSSFeed::serverURL @ " HTTP/1.0\r\n"
%data = %data @ "Host: " @ $RSSFeed::servername @ 
%data = %data @ "User-Agent: " @ $RSSFeed::userAgent @ 

// Send the command to the connected server
%thisTCPObj.send(%data);
```

Copyright © GarageGames, LLC. All Rights Reserved.
TerrainBlock Class Reference

[Terrain]

Represent a terrain object in a Torque 3D level. More...

Inheritance diagram for TerrainBlock:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>exportHeightMap</code> (string filename,[string format])</td>
<td>export the terrain block's heightmap to a bitmap file (default: png)</td>
</tr>
<tr>
<td>bool</td>
<td><code>exportLayerMaps</code> (string filePrefix,[string format])</td>
<td>export the terrain block's layer maps to bitmap files (default: png)</td>
</tr>
<tr>
<td>int</td>
<td><code>import</code> (String terrainName, String heightMap, F32 metersPerPixel, F32 heightScale, String materials, String opacityLayers)</td>
<td></td>
</tr>
<tr>
<td>bool</td>
<td><code>save</code> (string fileName)</td>
<td>Saves the terrain block's terrain file to the specified fileName.</td>
</tr>
</tbody>
</table>
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>createNew</td>
<td>TerrainBlock.create( String terrainName, U32 resolution, String materialName, bool genNoise ).</td>
</tr>
</tbody>
</table>

## Misc

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>baseTexSize</td>
<td>Size of base texture size per meter.</td>
</tr>
<tr>
<td>bool</td>
<td>castShadows</td>
<td>Allows the terrain to cast shadows onto itself and other objects.</td>
</tr>
<tr>
<td>int</td>
<td>lightMapSize</td>
<td>Light map dimensions in pixels.</td>
</tr>
<tr>
<td>int</td>
<td>screenError</td>
<td>Not yet implemented.</td>
</tr>
<tr>
<td>float</td>
<td>squareSize</td>
<td>Indicates the spacing between points on the XY plane on the terrain.</td>
</tr>
</tbody>
</table>

## Media

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filename</td>
<td>terrainFile</td>
<td>The source terrain data file.</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td><code>debugRender</code></td>
<td>Triggers debug rendering of terrain cells.</td>
</tr>
<tr>
<td>static bool</td>
<td><code>isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td><code>isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>

Detailed Description

Represent a terrain object in a Torque 3D level.

Example:

```java
new TerrainBlock(theTerrain)
{
    terrainFile = "art/terrains/Deathball_Desert_0.ter";
    squareSize = "2";
    tile = "0";
    baseTexSize = "1024";
    screenError = "16";
    position = "-1024 -1024 179.978";
    rotation = "1 0 0 0";
    scale = "1 1 1";
    isRenderEnabled = "true";
    canSaveDynamicFields = "1";
};
```

See also:

TerrainMaterial
### Member Function Documentation

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bool TerrainBlock::exportHeightMap(string filename)</code></td>
<td>export the terrain block's heightmap to a bitmap file (default: png)</td>
</tr>
<tr>
<td><code>bool TerrainBlock::exportLayerMaps(string filePrefix)</code></td>
<td>export the terrain block's layer maps to bitmap files (default: png)</td>
</tr>
<tr>
<td><code>int TerrainBlock::import(String terrainName, String heightMap, F32 metersPerPixel, F32 heightScale, String materials, String opacityLayers)</code></td>
<td>Saves the terrain block's terrain file to the specified file name.</td>
</tr>
<tr>
<td><code>bool TerrainBlock::save(string fileName)</code></td>
<td></td>
</tr>
</tbody>
</table>

**Parameters:**

- `fileName` Name and path of file to save terrain data to.

**Returns:**

True if file save was successful, false otherwise
### Member Data Documentation

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>TerrainBlock::baseTexSize</td>
<td>Size of base texture size per meter.</td>
</tr>
<tr>
<td>bool</td>
<td>TerrainBlock::castShadows</td>
<td>Allows the terrain to cast shadows onto itself and other objects.</td>
</tr>
<tr>
<td>int</td>
<td>TerrainBlock::createNew</td>
<td>TerrainBlock.create( String terrainName, U32 resolution, String materialName, bool genNoise ).</td>
</tr>
<tr>
<td>int</td>
<td>TerrainBlock::lightMapSize</td>
<td>Light map dimensions in pixels.</td>
</tr>
<tr>
<td>int</td>
<td>TerrainBlock::screenError</td>
<td>Not yet implemented.</td>
</tr>
<tr>
<td>float</td>
<td>TerrainBlock::squareSize</td>
<td>Indicates the spacing between points on the XY plane on the terrain.</td>
</tr>
<tr>
<td>filename</td>
<td>TerrainBlock::terrainFile</td>
<td></td>
</tr>
</tbody>
</table>
The source terrain data file.

Copyright © GarageGames, LLC. All Rights Reserved.
TerrainMaterial Class Reference
[Miscellaneous]

The TerrainMaterial class organizes the material settings for a single terrain material layer. More...

Inheritance diagram for TerrainMaterial:

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>detailDistance</td>
<td>Changes how far camera can see the detail map rendering on the material.</td>
</tr>
<tr>
<td>filename</td>
<td>detailMap</td>
<td>Detail map for the material.</td>
</tr>
<tr>
<td>float</td>
<td>detailSize</td>
<td>Used to scale the detail map to the material square.</td>
</tr>
<tr>
<td>float</td>
<td>detailStrength</td>
<td>Exponentially sharpens or lightens the detail map rendering on the material.</td>
</tr>
<tr>
<td>filename</td>
<td>diffuseMap</td>
<td>Base texture for the material.</td>
</tr>
<tr>
<td>float</td>
<td>diffuseSize</td>
<td>Used to scale the diffuse map to the material square.</td>
</tr>
<tr>
<td>filename</td>
<td>normalMap</td>
<td>Bump map for the material.</td>
</tr>
<tr>
<td>float</td>
<td>parallaxScale</td>
<td>Used to scale the height from the normal map to give some self occlusion effect (aka parallax) to the terrain material.</td>
</tr>
<tr>
<td>bool</td>
<td>useSideProjection</td>
<td>Makes that terrain material project along the sides of steep slopes instead of projected downwards.</td>
</tr>
</tbody>
</table>
Detailed Description

The TerrainMaterial class organizes the material settings for a single terrain material layer.

Note:

You should not be creating TerrainMaterials by hand in code. All TerrainMaterials should be created in the editors, as intended by the system.

Example:

```java
// Created by the Terrain Painter tool in the World Editor
new TerrainMaterial()
{
    internalName = "grass1";
    diffuseMap = "art/terrains/Test/grass1"
    detailMap = "art/terrains/Test/grass1_d"
    detailSize = "10"
    isManaged = "1"
    detailBrightness = "1"
    Enabled = "1"
    diffuseSize = "200"
};
```

See also:

Materials
## Member Data Documentation

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>TerrainMaterial::detailDistance</code></td>
<td>Changes how far camera can see the detail map rendering on the material.</td>
</tr>
<tr>
<td>filename</td>
<td><code>TerrainMaterial::detailMap</code></td>
<td>Detail map for the material.</td>
</tr>
<tr>
<td>float</td>
<td><code>TerrainMaterial::detailSize</code></td>
<td>Used to scale the detail map to the material square.</td>
</tr>
<tr>
<td>float</td>
<td><code>TerrainMaterial::detailStrength</code></td>
<td>Exponentially sharpens or lightens the detail map rendering on the material.</td>
</tr>
<tr>
<td>filename</td>
<td><code>TerrainMaterial::diffuseMap</code></td>
<td>Base texture for the material.</td>
</tr>
<tr>
<td>float</td>
<td><code>TerrainMaterial::diffuseSize</code></td>
<td>Used to scale the diffuse map to the material square.</td>
</tr>
<tr>
<td>filename</td>
<td><code>TerrainMaterial::normalMap</code></td>
<td></td>
</tr>
</tbody>
</table>
Bump map for the material.

**float TerrainMaterial::parallaxScale**

Used to scale the height from the normal map to give some self occlusion effect (aka parallax) to the terrain material.

**bool TerrainMaterial::useSideProjection**

Makes that terrain material project along the sides of steep slopes instead of projected downwards.

Copyright © GarageGames, LLC. All Rights Reserved.
TimeOfDay Class Reference

[Miscellaneous]

Environmental object that triggers a day/night cycle in level. More...

Inheritance diagram for TimeOfDay:

List of all members.
Public Member Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>addTimeOfDayEvent</td>
<td>float elevation, string identifier</td>
</tr>
<tr>
<td>animate</td>
<td>float elevation, float degreesPerSecond</td>
</tr>
<tr>
<td>setDayLength</td>
<td>float seconds</td>
</tr>
<tr>
<td>setPlay</td>
<td>bool enabled</td>
</tr>
<tr>
<td>setTimeOfDay</td>
<td>float time</td>
</tr>
</tbody>
</table>
## Public Attributes

### TimeOfDay

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>axisTilt</td>
<td>The angle in degrees between global equator and tropic.</td>
</tr>
<tr>
<td>float</td>
<td>azimuthOverride</td>
<td></td>
</tr>
<tr>
<td>float</td>
<td>dayLength</td>
<td>The length of a virtual day in real world seconds.</td>
</tr>
<tr>
<td>float</td>
<td>dayScale</td>
<td>Scalar applied to time that elapses while the sun is up.</td>
</tr>
<tr>
<td>float</td>
<td>nightScale</td>
<td>Scalar applied to time that elapses while the sun is down.</td>
</tr>
<tr>
<td>bool</td>
<td>play</td>
<td>True when the TimeOfDay object is operating.</td>
</tr>
<tr>
<td>float</td>
<td>startTime</td>
<td></td>
</tr>
<tr>
<td>float</td>
<td>time</td>
<td>Current time of day.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Environmental object that triggers a day/night cycle in level.

Note:

*TimeOfDay* only works in Advanced Lighting with a Sub object or *ScatterSky*

Example:

```javascript
new TimeOfDay(tod)
{
    axisTilt = "23.44";
    dayLength = "120";
    startTime = "0.15";
    time = "0.15";
    play = "0";
    azimuthOverride = "572.958";
    dayScale = "1";
    nightScale = "1.5";
    position = "598.399 550.652 196.297";
    rotation = "1 0 0 0";
    scale = "1 1 1";
    canSave = "1";
    canSaveDynamicFields = "1";
}
```
### Member Function Documentation

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void TimeOfDay::addTimeOfDayEvent(float elevation, string identifier)</code></td>
<td>Adds a time of day event.</td>
</tr>
<tr>
<td><code>void TimeOfDay::animate(float elevation, float degreesPerSecond)</code></td>
<td>Animates the time of day.</td>
</tr>
<tr>
<td><code>void TimeOfDay::setDayLength(float seconds)</code></td>
<td>Sets the day length.</td>
</tr>
<tr>
<td><code>void TimeOfDay::setPlay(bool enabled)</code></td>
<td>Enables or disables the play of time of day.</td>
</tr>
<tr>
<td><code>void TimeOfDay::setTimeOfDay(float time)</code></td>
<td>Sets the current time of day.</td>
</tr>
</tbody>
</table>
**Member Data Documentation**

**float** `TimeOfDay::axisTilt`

The angle in degrees between global equator and tropic.

**float** `TimeOfDay::azimuthOverride`

**float** `TimeOfDay::dayLength`

The length of a virtual day in real world seconds.

**float** `TimeOfDay::dayScale`

Scalar applied to time that elapses while the sun is up.

**float** `TimeOfDay::nightScale`

Scalar applied to time that elapses while the sun is down.

**bool** `TimeOfDay::play`

True when the `TimeOfDay` object is operating.

**float** `TimeOfDay::startTime`

**float** `TimeOfDay::time`

Current time of day.
A Trigger is a volume of space that initiates script callbacks when objects pass through the Trigger. More...

Inheritance diagram for Trigger:

```
SimObject
   ↓
NetObject
   ↓
SceneObject
   ↓
GameBase
   ↓
Trigger
```

List of all members.
Public Member Functions

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int getNumObjects ()</td>
<td>Get the number of objects that are within the Trigger's bounds.</td>
</tr>
<tr>
<td>int getObject(int index)</td>
<td>Retrieve the requested object that is within the Trigger's bounds.</td>
</tr>
</tbody>
</table>

Callbacks

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void onAdd(int objectId)</td>
<td>Called when the Trigger is being created.</td>
</tr>
<tr>
<td>void onRemove(int objectId)</td>
<td>Called just before the Trigger is deleted.</td>
</tr>
</tbody>
</table>
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>enterCommand</td>
<td>The command to execute when an object enters this trigger. Object id stored in %obj. Maximum 1023 characters.</td>
</tr>
<tr>
<td>string</td>
<td>leaveCommand</td>
<td>The command to execute when an object leaves this trigger. Object id stored in %obj. Maximum 1023 characters.</td>
</tr>
<tr>
<td>floatList</td>
<td>polyhedron</td>
<td>Defines a non-rectangular area for the trigger.</td>
</tr>
<tr>
<td>string</td>
<td>tickCommand</td>
<td>The command to execute while an object is inside this trigger. Maximum 1023 characters.</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>Attribute Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>renderTriggers</td>
<td>Forces all Trigger's to render.</td>
</tr>
</tbody>
</table>
Detailed Description

A Trigger is a volume of space that initiates script callbacks when objects pass through the Trigger.

TriggerData provides the callbacks for the Trigger when an object enters, stays inside or leaves the Trigger's volume.

See also:
  TriggerData
Member Function Documentation

```cpp
int Trigger::getNumObjects()
```

Get the number of objects that are within the Trigger's bounds.

**See also:**
`getObject()`

```cpp
int Trigger::getObject(int index)
```

Retrieve the requested object that is within the Trigger's bounds.

**Parameters:**

- `index` Index of the object to get (range is 0 to `getNumObjects()` - 1)

**Returns:**

The SimObjectID of the object, or -1 if the requested index is invalid.

**See also:**
`getNumObjects()`

```cpp
void Trigger::onAdd(int objectId)
```

Called when the Trigger is being created.

**Parameters:**

- `objectId` the object id of the Trigger being created

```cpp
void Trigger::onRemove(int objectId)
```
Called just before the Trigger is deleted.

**Parameters:**

- `objectId` the object id of the Trigger being deleted
Member Data Documentation

string Trigger::enterCommand

The command to execute when an object enters this trigger. Object id stored in %obj. Maximum 1023 characters.

string Trigger::leaveCommand

The command to execute when an object leaves this trigger. Object id stored in %obj. Maximum 1023 characters.

floatList Trigger::polyhedron

Defines a non-rectangular area for the trigger.

Rather than the standard rectangular bounds, this optional parameter defines a quadrilateral trigger area. The quadrilateral is defined as a corner point followed by three vectors representing the edges extending from the corner.

string Trigger::tickCommand

The command to execute while an object is inside this trigger. Maximum 1023 characters.
TriggerData Class Reference

[Game Objects]

Defines shared properties for Trigger objects. More...

Inheritance diagram for TriggerData:

```
   SimObject
      |
      v
   SimDataBlock
      |
      v
   GameBaseData
      |
      v
   TriggerData
```

List of all members.
# Callbacks

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool <code>clientSide</code></td>
<td>Forces <code>Trigger</code> callbacks to only be called on clients.</td>
</tr>
<tr>
<td>int <code>tickPeriodMS</code></td>
<td>Time in milliseconds between calls to <code>onTickTrigger()</code> while at least one object is within a Trigger's bounds.</td>
</tr>
</tbody>
</table>

```c
void onEnterTrigger (Trigger trigger, GameBase obj)
```

Called when an object enters the volume of the `Trigger` instance using this `TriggerData`.

```c
void onLeaveTrigger (Trigger trigger, GameBase obj)
```

Called when an object leaves the volume of the `Trigger` instance using this `TriggerData`.

```c
void onTickTrigger (Trigger trigger)
```

Called every `tickPeriodMS` number of milliseconds (as specified in the `TriggerData`) whenever one or more objects are inside the volume of the trigger.
Detailed Description

Defines shared properties for Trigger objects.

The primary focus of the TriggerData datablock is the callbacks it provides when an object is within or leaves the Trigger bounds.

See also:

  Trigger.
Member Function Documentation

```cpp
void TriggerData::onEnterTrigger (Trigger trigger, GameBase obj )
```

Called when an object enters the volume of the Trigger instance using this TriggerData.

**Parameters:**

- `trigger` the Trigger instance whose volume the object entered
- `obj` the object that entered the volume of the Trigger instance

```cpp
void TriggerData::onLeaveTrigger (Trigger trigger, GameBase obj )
```

Called when an object leaves the volume of the Trigger instance using this TriggerData.

**Parameters:**

- `trigger` the Trigger instance whose volume the object left
- `obj` the object that left the volume of the Trigger instance

```cpp
void TriggerData::onTickTrigger (Trigger trigger )
```

Called every tickPeriodMS number of milliseconds (as specified in the TriggerData) whenever one or more objects are inside the volume of the trigger.

The Trigger has methods to retrieve the objects that are within the Trigger's bounds if you want to do something with them in this callback.
Parameters:

trigger the Trigger instance whose volume the object is inside

See also:

tickPeriodMS
Trigger::getNumObjects()
Trigger::getObject()
Member Data Documentation

**bool TriggerData::clientSide**

Forces Trigger callbacks to only be called on clients.

**int TriggerData::tickPeriodMS**

Time in milliseconds between calls to onTickTrigger() while at least one object is within a Trigger's bounds.

**See also:**

onTickTrigger()
TSForestItemData Class Reference

[Forest]

Concrete implementation of ForestItemData which loads and renders dts format shapeFiles. More...

Inheritance diagram for TSForestItemData:

```
SimObject
    ▲
  SimDataBlock
    ▲
ForestItemData
    ▲
TSForestItemData
          [legend]
```

List of all members.
Detailed Description

Concrete implementation of ForestItemData which loads and renders dts format shapeFiles.
TSShapeConstructor Class Reference
[Game Objects]

An object used to modify a DTS or COLLADA shape model after it has been loaded by Torque. More...

Inheritance diagram for TSShapeConstructor:

List of all members.
# Public Member Functions

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Function Name</th>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>addCollisionDetail</code></td>
<td>(int size, string type, string target, int depth=4, float merge=30, float concavity=30, int maxVerts=32, float boxMaxError=0, float sphereMaxError=0, float capsuleMaxError=0)</td>
<td>Autofit a mesh primitive or set of convex hulls to the shape geometry. Hulls may optionally be converted to boxes, spheres and/or capsules based on their volume.</td>
</tr>
<tr>
<td>int</td>
<td><code>addImposter</code></td>
<td>(int size, int equatorSteps, int polarSteps, int dl, int dim, bool includePoles, float polarAngle)</td>
<td>Add (or edit) an imposter detail level to the shape.</td>
</tr>
<tr>
<td>bool</td>
<td><code>addMesh</code></td>
<td>(string meshName, string srcShape, string srcMesh)</td>
<td>Add geometry from another DTS or DAE shape file into this shape.</td>
</tr>
<tr>
<td>bool</td>
<td><code>addNode</code></td>
<td>(string name, string parentName, TransformF txfm=TransformF::Identity, bool isWorld=false)</td>
<td>Add a new node.</td>
</tr>
<tr>
<td>bool</td>
<td><code>addPrimitive</code></td>
<td>(string meshName, string type, string params, TransformF txfm, string nodeName)</td>
<td>Add a new mesh primitive to the shape.</td>
</tr>
<tr>
<td>bool</td>
<td><code>addSequence</code></td>
<td>(string source, string name, int start=0, int end=-1, bool padRot=true, bool padTrans=false)</td>
<td>Add a new sequence to the shape.</td>
</tr>
<tr>
<td>bool</td>
<td><code>addTrigger</code></td>
<td>(string name, int keyframe, int state)</td>
<td>Add a new trigger to the sequence.</td>
</tr>
<tr>
<td>void</td>
<td><code>dumpShape</code></td>
<td>(string filename=&quot;&quot;)</td>
<td>Dump the shape hierarchy to the console or to a file. Useful for reviewing the result of a series of construction commands.</td>
</tr>
<tr>
<td>Function/Property</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Box3F <code>getBounds()</code></td>
<td>Get the bounding box for the shape.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>int <code>getDetailLevelCount</code>()</td>
<td>Get the total number of detail levels in the shape.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>int <code>getDetailLevelIndex(int size)</code></td>
<td>Get the index of the detail level with a given size.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>string <code>getDetailLevelName(int index)</code></td>
<td>Get the name of the indexed detail level.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>int <code>getDetailLevelSize(int index)</code></td>
<td>Get the size of the indexed detail level.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>int <code>getImposterDetailLevel()</code></td>
<td>Get the index of the imposter (auto-billboard) detail level (if any).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>string <code>getImposterSettings(int index)</code></td>
<td>Get the settings used to generate imposters for the indexed detail level.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>int <code>getMeshCount(string name)</code></td>
<td>Get the number of meshes (detail levels) for the specified object.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>string <code>getMeshMaterial(string name)</code></td>
<td>Get the name of the material attached to a mesh. Note that only the first material used by the mesh is returned.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>string <code>getMeshName(string name, int index)</code></td>
<td>Get the name of the indexed mesh (detail level) for the specified object.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>int <code>getMeshSize(string name, int index)</code></td>
<td>Get the detail level size of the indexed mesh for the specified object.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>string <code>getMeshType(string name)</code></td>
<td>Get the display type of the mesh.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>int <code>getNodeChildCount(string name)</code></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>getNodeChildName (string name, int index)</code></td>
<td>Get the number of children of this node.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>getNodeCount ()</code></td>
<td>Get the total number of nodes in the shape.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>getNodeIndex (string name)</code></td>
<td>Get the index of the node.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>getNodeName (int index)</code></td>
<td>Get the name of the indexed node.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>getNodeObjectCount (string name)</code></td>
<td>Get the number of geometry objects attached to this node.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>getNodeObjectName (string name, int index)</code></td>
<td>Get the name of the indexed object.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>getNodeParentName (string name)</code></td>
<td>Get the name of the node's parent. If the node has no parent (ie. it is at the root level), return an empty string.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>getNodeTransform (string name, bool isWorld=false)</code></td>
<td>Get the base (ie. not animated) transform of a node.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>getObjectCount ()</code></td>
<td>Get the total number of objects in the shape.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>getObjectIndex (string name)</code></td>
<td>Get the index of the first object with the given name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>getObjectName (int index)</code></td>
<td>Get the name of the indexed object.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>getObjectNode (string name)</code></td>
<td>Get the name of the node this object is attached to.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>getSequenceBlend (string name)</code></td>
<td>Get information about blended sequences.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>getSequenceCount ()</code></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Get the total number of sequences in the shape.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bool getSequenceCyclic (string name)</code></td>
<td>Check if this sequence is cyclic (looping).</td>
</tr>
<tr>
<td><code>int getSequenceFrameCount (string name)</code></td>
<td>Get the number of keyframes in the sequence.</td>
</tr>
<tr>
<td><code>string getSequenceGroundSpeed (string name)</code></td>
<td>Get the ground speed of the sequence.</td>
</tr>
<tr>
<td><code>int getSequenceIndex (string name)</code></td>
<td>Find the index of the sequence with the given name.</td>
</tr>
<tr>
<td><code>string getSequenceName (int index)</code></td>
<td>Get the name of the indexed sequence.</td>
</tr>
<tr>
<td><code>float getSequencePriority (string name)</code></td>
<td>Get the priority setting of the sequence.</td>
</tr>
<tr>
<td><code>string getSequenceSource (string name)</code></td>
<td>Get information about where the sequence data came from.</td>
</tr>
<tr>
<td><code>int getTargetCount ()</code></td>
<td>Get the number of materials in the shape.</td>
</tr>
<tr>
<td><code>string getTargetName (int index)</code></td>
<td>Get the name of the indexed shape material.</td>
</tr>
<tr>
<td><code>string getTrigger (string name, int index)</code></td>
<td>Get information about the indexed trigger.</td>
</tr>
<tr>
<td><code>int getTriggerCount (string name)</code></td>
<td>Get the number of triggers in the specified sequence.</td>
</tr>
<tr>
<td><code>void notifyShapeChanged ()</code></td>
<td>Notify game objects that this shape file has changed, allowing them to update internal data if needed.</td>
</tr>
<tr>
<td><code>bool removeDetailLevel (int index)</code></td>
<td>Remove the detail level (including all meshes in the detail level).</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>bool removeImposter ()</code></td>
<td>Remove the imposter detail level (if any) from the shape.</td>
</tr>
<tr>
<td><code>bool removeMesh (string name)</code></td>
<td>Remove a mesh from the shape.</td>
</tr>
<tr>
<td><code>bool removeNode (string name)</code></td>
<td>Remove a node from the shape.</td>
</tr>
<tr>
<td><code>bool removeObject (string name)</code></td>
<td>Remove an object (including all meshes for that object) from the shape.</td>
</tr>
<tr>
<td><code>bool removeSequence (string name)</code></td>
<td>Remove the sequence from the shape.</td>
</tr>
<tr>
<td><code>bool removeTrigger (string name, int keyframe, int state)</code></td>
<td>Remove a trigger from the sequence.</td>
</tr>
<tr>
<td><code>bool renameDetailLevel (string oldName, string newName)</code></td>
<td>Rename a detail level.</td>
</tr>
<tr>
<td><code>bool renameNode (string oldName, string newName)</code></td>
<td>Rename a node.</td>
</tr>
<tr>
<td><code>bool renameObject (string oldName, string newName)</code></td>
<td>Rename an object.</td>
</tr>
<tr>
<td><code>bool renameSequence (string oldName, string newName)</code></td>
<td>Rename a sequence.</td>
</tr>
<tr>
<td><code>void saveShape (string filename)</code></td>
<td>Save the shape (with all current changes) to a new DTS file.</td>
</tr>
<tr>
<td><code>bool setBounds (Box3F bbox)</code></td>
<td>Set the shape bounds to the given bounding box.</td>
</tr>
<tr>
<td><code>int setDetailLevelSize (int index, int newSize)</code></td>
<td>Change the size of a detail level.</td>
</tr>
<tr>
<td><code>bool setMeshMaterial (string meshName, string matName)</code></td>
<td></td>
</tr>
</tbody>
</table>
Set the name of the material attached to the mesh.

<table>
<thead>
<tr>
<th>Function</th>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bool setMeshSize</code></td>
<td>(string name, int size)</td>
<td>Change the detail level size of the named mesh.</td>
</tr>
<tr>
<td><code>bool setMeshType</code></td>
<td>(string name, string type)</td>
<td>Set the display type for the mesh.</td>
</tr>
<tr>
<td><code>bool setNodeParent</code></td>
<td>(string name, string parentName)</td>
<td>Set the parent of a node.</td>
</tr>
<tr>
<td><code>bool setNodeTransform</code></td>
<td>(string name, TransformF txfm, bool isWorld=false)</td>
<td>Set the base transform of a node. That is, the transform of the node when in the root (not-animated) pose.</td>
</tr>
<tr>
<td><code>bool setObjectNode</code></td>
<td>(string objName, string nodeName)</td>
<td>Set the node an object is attached to.</td>
</tr>
<tr>
<td><code>bool setSequenceBlend</code></td>
<td>(string name, bool blend, string blendSeq, int blendFrame)</td>
<td>Mark a sequence as a blend or non-blend.</td>
</tr>
<tr>
<td><code>bool setSequenceCyclic</code></td>
<td>(string name, bool cyclic)</td>
<td>Mark a sequence as cyclic or non-cyclic.</td>
</tr>
<tr>
<td><code>bool setSequenceGroundSpeed</code></td>
<td>(string name, Point3F transSpeed, Point3F rotSpeed=Point3F::Zero)</td>
<td>Set the translation and rotation ground speed of the sequence.</td>
</tr>
<tr>
<td><code>bool setSequencePriority</code></td>
<td>(string name, float priority)</td>
<td>Set the sequence priority.</td>
</tr>
</tbody>
</table>

```cpp
void writeChangeSet ()
```

Write the current change set to a `TSShapeConstructor` script file. The name of the script file is the same as the model, but with `.cs` extension. eg. `myShape.cs` for `myShape.dts` or `myShape.dae`.

**Callbacks**
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>**void **onLoad ()</td>
<td>Called immediately after the DTS or DAE file has been loaded; before the shape data is available to any other object (<strong>StaticShape</strong>, <strong>Player</strong> etc). This is where you should put any post-load commands to modify the shape in-memory such as addNode, renameSequence etc.</td>
</tr>
<tr>
<td>**void **onUnload ()</td>
<td>Called when the DTS or DAE resource is flushed from memory. Not normally required, but may be useful to perform cleanup.</td>
</tr>
</tbody>
</table>
### Public Attributes

#### Collada

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>adjustCenter</code></td>
<td>Translate COLLADA model on import so the origin is at the center. No effect for DTS files.</td>
</tr>
<tr>
<td>bool</td>
<td><code>adjustFloor</code></td>
<td>Translate COLLADA model on import so origin is at the (Z axis) bottom of the model. No effect for DTS files.</td>
</tr>
<tr>
<td>string</td>
<td><code>alwaysImport</code></td>
<td>TAB separated patterns of nodes to import even if in <code>neverImport</code> list. No effect for DTS files.</td>
</tr>
<tr>
<td>string</td>
<td><code>alwaysImportMesh</code></td>
<td>TAB separated patterns of meshes to import even if in <code>neverImportMesh</code> list. No effect for DTS files.</td>
</tr>
<tr>
<td>bool</td>
<td><code>forceUpdateMaterials</code></td>
<td>Forces update of the materials.cs file in the same folder as the COLLADA (.dae) file, even if Materials already exist. No effect for DTS files.</td>
</tr>
<tr>
<td>bool</td>
<td><code>ignoreNodeScale</code></td>
<td>Ignore <code>&lt;scale&gt;</code> elements inside COLLADA <code>&lt;node&gt;</code>s. No effect for DTS files.</td>
</tr>
<tr>
<td>TSShapeConstructorLodType</td>
<td><code>lodType</code></td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Control how the COLLADA (.dae) importer interprets LOD in the model. No effect for DTS files.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>string matNamePrefix</td>
<td>Prefix to apply to all material map names in the COLLADA (.dae) file. No effect for DTS files.</td>
<td></td>
</tr>
<tr>
<td>string neverImport</td>
<td>TAB separated patterns of nodes to ignore on loading. No effect for DTS files.</td>
<td></td>
</tr>
<tr>
<td>string neverImportMesh</td>
<td>TAB separated patterns of meshes to ignore on loading. No effect for DTS files.</td>
<td></td>
</tr>
<tr>
<td>int singleDetailSize</td>
<td>Sets the detail size when LodType is set to SingleSize. No effect otherwise, and no effect for DTS files.</td>
<td></td>
</tr>
<tr>
<td>float unit</td>
<td>Override the &lt;unit&gt; element in the COLLADA (.dae) file. No effect for DTS files.</td>
<td></td>
</tr>
<tr>
<td>TSShapeConstructorUpAxis</td>
<td>Override the &lt;up_axis&gt; element in the COLLADA (.dae) file. No effect for DTS files.</td>
<td></td>
</tr>
<tr>
<td>Media</td>
<td></td>
<td></td>
</tr>
<tr>
<td>filename baseShape</td>
<td>Specifies the path to the DTS or DAE file to be operated on by this object.</td>
<td></td>
</tr>
</tbody>
</table>
## Sequences

<table>
<thead>
<tr>
<th>filename</th>
<th>sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Legacy method of adding sequences to a DTS or DAE shape after loading.</td>
</tr>
</tbody>
</table>
Detailed Description

An object used to modify a DTS or COLLADA shape model after it has been loaded by Torque.

TSShapeConstructor is a special object used to modify a DTS or COLLADA shape model after it has been loaded by Torque, but before it is used by any other object.

It is often used to share animations from DSQ files between shapes with a common skeleton.

It may also be used to 'Torquify' a model that is missing the nodes and/or sequences required to function as a particular Torque object. A model used for a Player character for example should have an eye and a cam node, but these might not be present in a model not specifically created for Torque. TSShapeConstructor allows the missing nodes to be added and positioned so that the shape does not need to be re-worked or re-exported by an artist.

Note:

In T3D, TSShapeConstructor is no longer a SimDataBlock (like in TGE and TGEA), but a SimObject instead. This means that all declarations of TSShapeConstructor should use the singleton keyword instead of the datablock keyword.

TSShapeConstructor also includes features to aid in loading COLLADA models, such as allowing the <up_axis> and <unit> elements to be overridden, and can also apply a user specified prefix to the names of COLLADA materials as shown below. Prefixing material names is useful to avoid name clashes, particularly for 3D apps like Google SketchUp that export models with generic material names like "material0". These options are most easily accessed using the COLLADA import gui, which will be displayed automatically the first time a COLLADA model is imported into Torque.
Settings from the import gui are automatically saved to a `TSShapeConstructor` script in the same folder as the model.

To create your own `TSShapeConstructor` object, simply create a TorqueScript file in the same folder as your DTS or COLLADA model, with the same filename but .cs extension. For example, if your model file was called `myShape.dts`, you would create a file called `myShape.cs`. Some example appear below:

```plaintext
singleton TSShapeConstructor(MyShapeDae)
{
    baseShape = "./myShape.dae";
    upAxis = "DEFAULT"; // use <up_axis
    unit = "1.0"; // override <unit>
    matNamePrefix = ""; // no prefix
};

singleton TSShapeConstructor(MyShape2Dae)
{
```
The name of the `TSShapeConstructor` object (MyShapeDae and MyShape2Dae in the code samples above) is up to you, but you should choose a name that does not conflict with other objects or datablocks. A common convention for `TSShapeConstructor` objects is the name of the shape file. eg. MyShapeDae for a file called myshape.dae.

When Torque loads a DTS (.dts) or COLLADA (.dae) file, it first looks in the same folder for a TorqueScript file with the same filename (but .cs extension) in order to create the `TSShapeConstructor` object. Such scripts are executed automatically by Torque 3D, so there is no need to manually call exec("myShape.cs") from another script. Also, you should avoid adding other object and datablock declarations to this script because it will be executed every time the model is loaded, which may cause unexpected results if the datablocks already exist.

After Torque has loaded the model from the DTS or COLLADA file, it executes the `TSShapeConstructor` onLoad method to apply the desired set of changes to the shape. It should be noted that the changes are applied to the loaded model in memory rather than to the DTS or COLLADA file itself. This means the model can be re-exported to DTS or COLLADA without overwriting the `TSShapeConstructor` changes. `TSShapeConstructor` should be thought of as a post-export processing step, and is intended to be used alongside existing object and datablock setups.

Note that DSQ sequences may still be specified within the `TSShapeConstructor` object as normal:

```
singleton TSShapeConstructor( PlayerDts )
{
  baseShape = "./myShape2.dae";
  upAxis = "Y_AXIS";  // override <
  unit = -1;         // use <unit>
  matNamePrefix = "myShape2_"; // prefix all
}
```
baseShape = "./player.dts";
sequence = "/art/shapes/actors/animations/player_root.dsq	root
sequence = "/art/shapes/actors/animations/player_forward.dsq	run

function PlayerDts::onLoad( %this )
{
    %this.addCollisionDetail( -1, "box", "bounds"
    %this.addSequence( "walk", "walk_short", 1
    %this.addTrigger( "walk_short", 3, 4 );
    %this.renameSequence( "walk", "walk_long"
    %this.renameNode( "Bip Pelvis", "pelvis" )
    %this.addNode( "mount5", "Bip L Hand" );
    %this.dumpShape();
}

Note that most of the features in TSShapeConstructor are far more easily accessible in the Shape Editor tool. This tool uses TSShapeConstructor 'under-the-hood' to edit the nodes, sequences and details of a shape, and the changes are saved to a TSShapeConstructor script object.
Shape Terminology

The following definitions should be understood before reading the TSShapeConstructor examples and function reference:

node

A node is a place-holder for transform (position and rotation) information. Nodes are arranged in a parent/child hierarchy allowing complex skeleton structures to be built up.

sequence

A sequence is a set of keyframed node transforms and object states (eg. visibility).

detail level

Level-of-Detail (LOD) is very important to maintain good rendering performance. When a shape is rendered, Torque automatically picks a single detail level based on that shape's on-screen size. Only the sub-meshes belonging to that detail level are rendered.

object

An object is a collection of meshes, each at a different detail level. Each object is attached to a certain node, and is rendered at that node's current transform.

mesh

A mesh is a piece of geometry, and may be a rigid body or a skin (vertex weighted mesh). Each mesh is associated with an object and a certain detail level.
Example 1: Adding a Collision Mesh To an Existing Shape

Imagine you have a model that you want to add to the scene as a `StaticShape`, but it is missing a collision mesh. `TSShapeConstructor` makes it simple to modify an existing DTS shape to add a collision (or line-of-sight collision) detail level.

First, define the `StaticShapeData` datablock as normal. Create a script called `myShape.cs` in the `art/datablocks` folder, and define the datablock:

```plaintext
datablock StaticShapeData( MyShapeData ) {
    category = "Misc";
    shapeFile = "art/shapes/myShape/myShape.dts"
};
```

We need to tell Torque to execute this script so add `exec("./myShape.cs");` to `art/datablocks/datablockExec.cs`.

Now we define the `TSShapeConstructor` object by creating a new script called `myShape.cs` in the `art/shapes/myShape` folder:

```plaintext
singleton TSShapeConstructor( MyShapeDts ) {
    baseShape = "~/art/shapes/myShape/myShape.dts"
};

function MyShapeDts::onLoad( %this ) {
    %this.addCollisionDetail( -1, "box", "boun
```

This script will add a box mesh with the same center and dimensions as the original model using the "Col" detail level at size -1. The
negative detail size means that the mesh will not be rendered in-game, and the use of the special "Col" name means that this mesh will be detected as a collision mesh by the Torque engine.

When a Torque mission is started, the following steps occur:

1. On the server, Torque executes
   art/datablocks/datablockExec.cs, which in turn executes
   art/datablocks/myShape.cs
2. Executing art/datablocks/myShape.cs causes the MyShapeData datablock to attempt to load the DTS file
3. Before loading the DTS file, Torque executes
   art/shapes/myShape/myShape.cs, which creates the
   TSShapeConstructor object
4. After loading the DTS file, Torque executes
   MyShapeDts::onLoad, which adds the collision detail level to the shape
5. The StaticShape object can now be added to the scene (it will appear in the world editor (in the "Misc" category under the Library/Scripted tab)
Example 2: Adding a Mesh From an Existing DTS File

The image below shows a boulder.dts shape in the Torque Show Tool Pro (TSTPro). The circled items indicate the geometry and material that will be copied into a different shape using TSShapeConstructor.

The following shows how to include the boulder1 mesh in another shape:

```
singleton TSShapeConstructor( TestShapeDts )
{
    baseShape = "~/art/shapes/rocks/rock1.dts"
}
```
function TestShapeDts::onLoad( %this )
{
    %this.addMesh( "test128", "~/art/shapes/ro
        %this.dumpShape();
}

The output of the dumpShape command is shown below:

Shape Hierarchy:

Details:
- detail128, Subtree 0, objectDetail 0, size 128
- detail2, Subtree 0, objectDetail 1, size 2
- collision-1, Subtree 0, objectDetail 2,

Subtrees:
- Subtree 0
  - Rock2 --> Object Rock with following
    --> Object test with following
    col-1 --> Object col with following

Sequences:

%Material list:
- material #0: 'rock2'.
- material #1: 'MossyRock02'.

Note that the new detail ("detail128"), object ("test") and material ("MossyRock02") have been added to the normal rock1.dts shape.
Example 3: Auto-loading animations

Instead of manually specifying all of the animations to load, it's easy to write some TorqueScript that will scan a folder for any matching animations and add them to the shape automatically. Imagine that we have the following shape (DTS) and sequence (DSQ) files:

- `art/shapes/actors/ForgeSoldier/ForgeSoldier.dts`
- `art/shapes/actors/animations/player_root.dsq`
- `art/shapes/actors/animations/player_run.dsq`
- `art/shapes/actors/animations/player_back.dsq`
- `art/shapes/actors/animations/player_side.dsq`
- `art/shapes/actors/animations/player_death0.dsq`

The following script will scan the `animations` folder and add the sequences to the shape.

```torquescript
singleton TSShapeConstructor( ForgeSoldierDts ) {
    baseShape = "~/art/shapes/actors/ForgeSoldier/ForgeSoldier.dts";
}

function ForgeSoldierDts::onLoad( %this ) {
    AddAnimations( %this );
}

function AddAnimations( %shape ) {
    // Scan for all player_*_.dsq files and add
    // Sequence names are taken from the base
    // player_run.dsq would add a sequence cal

    %filePatterns = "art/shapes/actors/animations/player_*_.dsq"
    %fullPath = findFirstFileMultiExpr( %filePatterns )

    if( %fullPath ) {
        %sequenceNames = split( %fullPath, "\"" )
        %sequenceNames = %sequenceNames[1]

        %shape.addAnimation( %sequenceNames, %fullPath );
    }
}
```
while ( %fullPath !$= '"' )
{
    // add this animation to the shape
    %fullPath = makeRelativePath( %fullPath
    %seq_name = strreplace( filebase( %full
    %shape.addSequence( %fullpath, %seq_name
    %fullPath = findNextFileMultiExpr( %fil

}
Example 4: Splitting COLLADA animations

Many COLLADA exporters do not support the `<animation_clip>` element, meaning that animated models imported into Torque appear to have only a single sequence containing all of the animations. **TSShapeConstructor** can be used to split this combined animation into individual sequences. This is most easily done using the Shape Editor tool, but can also be done manually as follows:

```plaintext
singleton TSShapeConstructor( PlayerModelDts
{

    baseShape = "~/art/shapes/collada/myPlayer.dae"

};

function PlayerModelDts::onLoad( %this )
{

    // Split animations from combined "ambient"
    %this.addSequence( "ambient", "root", 0, 1
    %this.addSequence( "ambient", "run", 16, 4
    %this.addSequence( "ambient", "back", 41,
    %this.addSequence( "ambient", "side", 56,
    %this.addSequence( "ambient", "death0", 80

    // Remove combined "ambient" sequence => we
    %this.removeSequence( "ambient" );

```
Example 5: LOD using separate files

In the past, using LOD required the artist to export all detail levels into a single DTS file. Using TSShapeConstructor, we can combine separate model files together. In fact, we can even use the folder-scanning approach from Example 3 to automatically construct the shape detail levels using all of the model files in the folder!

Note that the detail level models must contain the same object name, and for skinned models, the skin must be applied to the same skeleton for this script to work.

Imagine that we have the following shape (DAE) files:

- `art/shapes/actors/soldier/soldier.dae` (the base model, highest detail level size=1024)
- `art/shapes/actors/soldier/soldier_512.dae`
- `art/shapes/actors/soldier/soldier_256.dae`
- `art/shapes/actors/soldier/soldier_64.dae`

```singleton TSShapeConstructor( SoldierDae )
{
    baseShape = "~/art/shapes/actors/soldier.dae"
};

function SoldierDae::onLoad( %this )
{
    %objName = %this.getObjectName( 0 );
    AddDetails( %this, %objName );
}

// Scan for all DAE files in the folder with // skinned mesh from that file to the input s // is taken from the end of the filename. eg. // level 30 to the shape.
// Note that this function can be in a differ```
multiple TSShapeConstructor objects...

function AddDetails( %shape, %objName )
{
    // Determine the base name of the input file
    // soldier.dae is 'soldier'.
    %inputBase = fileBase( %shape.baseShape );

    %filePatterns = filePath( %shape.baseShape );
    %fullPath = findFirstFileMultiExpr( %filePatterns );
    while (%fullPath !=$)
    {
        %fullPath = makeRelativePath( %fullPath );

        // Determine the detail size, then add
        %size = strreplace( fileBase( %fullPath );
        %shape.addMesh( %objName SPC %size, %fullPath);

        %fullPath = findNextFileMultiExpr( %filePatterns );
    }
}
Example 6: Add lights to the scene

Although most often used to modify a shape before it is used, TSShapeConstructor can also be used as a general purpose interface to a 3D shape. For example, a 3D modeling package could be used to layout positions for lights in the scene. On import, the shape hierarchy might look like this:

```
base01
  +-start01
    +-Lights
      +-lightA
      +-lightB
      +-lightC
      ...
```

The following code demonstrates how to create a TSShapeConstructor object on-demand in order to access the 3D shape data. This example adds lights to the current scene at position of the lightX nodes in the shape:

```plaintext
function AddLights( %obj )
{
  // Get a TSShapeConstructor for this object
  // utility functions to create one if it d
  %shapePath = ShapeEditor.getObjectShapeFile( %obj );
  %shape = ShapeEditor.findConstructor( %shapePath );
  if ( !isObject( %shape ) )
  %shape = ShapeEditor.createConstructor( %shapePath );
  if ( !isObject( %shape ) )
  {
    echo( "Failed to create TSShapeConstructor" );
    return;
  }
```
%objTransform = %obj.getTransform();

%lightsGroup = "Lights";
%count = %shape.getNodeChildCount( %lightsGroup)
for ( %i = 0; %i < %count; %i++ )
{
   // get node transform in object space,
   %child = %shape.getNodeChildName( %lightsGroup, %i)
   %txfm = %shape.getNodeTransform( %child)
   %txfm = MatrixMultiply( %objTransform, %txfm)

   // create a new light at the object node
   %light = new PointLight() {
      position = getWords( %txfm, 0, 2 );
      rotation = getWords( %txfm, 3, 6 );
   };
   MissionGroup.add( %light );
}

The original shape can be placed anywhere in the scene, then AddLights is called to create and place a PointLight at each node.
Example 7: Rigid-body Player Character

Using the addNode and addMesh functions, it is possible to create a rigid-body (ie. non-skinned) player model compatible with the default animations, completely from TorqueScript!

The default player skeleton node transforms were obtained by adding the following code to the TSShapeConstructor onLoad function for a shape that already contained the default skeleton:

```
%count = %this.getNodeCount();
for ( %i = 0; %i < %count; %i++ )
{
    %name = %this.getNodeName( %i );
    echo( "%shape.addNode("" @ %name @ "," , 
          %this.getNodeTransform( %name ) @ "")
}
```

The contents of the console can then be copied and pasted into a new script. The script below shows the player model creation process: first pick a dummy dts file (rock1.dts in this case), and delete its existing nodes and meshes. Then create the default player skeleton. Finally, some box meshes are added at certain nodes to build up a rigid-body player character.

```
// Create the default Torque skeleton (compat function CreateDefaultSkeleton( %shape )
{
    %shape.addNode( "Bip01 Pelvis", "", "Bip0
    %shape.addNode( "Bip01 Spine", "Bip0
    %shape.addNode( "Bip01 Spine1", "Bip0
    %shape.addNode( "Bip01 Spine2", "Bip0
    %shape.addNode( "Bip01 Neck", "Bip0
    %shape.addNode( "Bip01 Head", "Bip0
    %shape.addNode( "Eye", "Bip0
```
// Create a new box mesh at the specified node
function CreateBoxObject( %shape, %name, %node, %offset ) {
  if ( %offset $= "" )
    %offset = "0 0 0";

  %shape.addPrimitive( %name @ "2", "box", %offset );
}

// Create a simple (rigid) player model using
function CreateRigidPlayer( %shape )
{
  CreateBoxObject( %shape, "head",
  CreateBoxObject( %shape, "upper_torso",
  CreateBoxObject( %shape, "lower_torso",
  CreateBoxObject( %shape, "left_thigh",
  CreateBoxObject( %shape, "left_leg",
  CreateBoxObject( %shape, "left_foot",
  CreateBoxObject( %shape, "right_thigh",
  CreateBoxObject( %shape, "right_leg",
  CreateBoxObject( %shape, "right_foot",
  CreateBoxObject( %shape, "left_upper_arm",
  CreateBoxObject( %shape, "left_forearm",
  CreateBoxObject( %shape, "left_hand",
  CreateBoxObject( %shape, "right_upper_arm"
  CreateBoxObject( %shape, "right_forearm",
  CreateBoxObject( %shape, "right_hand",

  // create collision mesh
  %shape.addCollisionDetail( -1, "box", "box"

}

singleton TSShapeConstructor( TestModelDts )
{
  baseShape = "~/art/shapes/rocks/rock1.dts"
};

function TestModelDts::onLoad( %this )
{  
// remove the existing nodes and geometry
%count = %this.getObjectCount();
for ( %i = %count-1; %i >= 0; %i-- )
  %this.removeObject( %this.getObjectName( %i ) );
%count = %this.getNodeCount();
for ( %i = %count-1; %i >= 0; %i-- )
  %this.removeNode( %this.getNodeName( %i ) );

// create the player skeleton and rigid body parts
CreateDefaultSkeleton( %this );
CreateRigidPlayer( %this );

// now load the default sequences
%this.addSequence( "~/art/shapes/actors/animations/player_root.dsq" );
%this.addSequence( "~/art/shapes/actors/animations/player_forward.dsq" );
%this.dumpShape();
}

// The player model definition
singleton PlayerData( TestModelData : Default ) {
  renderFirstPerson = false;
  emap = true;

  // className = Armor;
  shapeFile = "~/art/shapes/rocks/rock1.dts"
};

PlayerDatasGroup.add( TestModelData );
This produces the following shape:

Shape Hierarchy:

Details:
  detail2, Subtree 0, objectDetail 0, size collision-1, Subtree 0, objectDetail 1,

Subtrees:
  Subtree 0
    Bip01 Pelvis -- Object col with fol
    Bip01 Spine --> Object lower_tors
      Bip01 Spine1
        Bip01 Spine2 --> Object upp
          Bip01 Neck
            Bip01 Head
              Eye --> Object hea
              Bip01 L Clavicle
                Bip01 L UpperArm -
                  Bip01 L Forearm
                    Bip01 L Hand
                Bip01 R Clavicle
                  Bip01 R UpperArm -
                    Bip01 R Forearm
                      Bip01 R Hand
                      Mount0
                      Mount1
          Bip01 L Thigh --> Object left_
            Bip01 L Calf -- Object lef
            Bip01 L Foot --> Object
              Ski0
            Bip01 R Thigh --> Object right.
Bip01 R Calf --> Object rig
Bip01 R Foot --> Object Ski1

Unlink
Cam

Sequences:
0: %Root (cyclic)
1: run (cyclic)
Member Function Documentation

```cpp
bool TSShapeConstructor::addCollisionDetail(int size,
                                          string type,
                                          string target,
                                          int depth = 4,
                                          float merge = 30,
                                          float concavity = 30,
                                          int maxVerts = 32,
                                          float boxMaxError = 0,
                                          float sphereMaxError =
                                          float capsuleMaxError =
                                      )
```

Autofit a mesh primitive or set of convex hulls to the shape geometry. Hulls may optionally be converted to boxes, spheres and/or capsules based on their volume.

**Parameters:**

- `size`: size for this detail level
- `type`: one of: box, sphere, capsule, 10-dop x, 10-dop y, 10-dop z, 18-dop, 26-dop, convex hulls. See the Shape Editor documentation for more details about these types.
- `target`: geometry to fit collision mesh(es) to; either "bounds" (for the whole shape), or the name of an object in the shape
- `depth`: maximum split recursion depth (hulls only)
- `merge`: volume % threshold used to merge hulls together (hulls only)
- `concavity`: volume % threshold used to detect concavity (hulls only)
- `maxVerts`: maximum number of vertices per hull (hulls only)
- `boxMaxError`: max % volume difference for a hull to be converted to a box (hulls only)
sphereMaxError  max % volume difference for a hull to be converted to a sphere (hulls only)
capsuleMaxError  max % volume difference for a hull to be converted to a capsule (hulls only)

Returns:
true if successful, false otherwise

Example:

```cpp
%this.addCollisionDetail( -1, "box", "bounds"
%this.addCollisionDetail( -1, "convex hull"
%this.addCollisionDetail( -1, "convex hull"
```

```cpp
int TSShapeConstructor::addImposter(int size,
    int equatorSteps,
    int polarSteps,
    int dl,
    int dim,
    bool includePoles,
    float polarAngle
)
```

Add (or edit) an imposter detail level to the shape.

If the shape already contains an imposter detail level, this command will simply change the imposter settings

Parameters:

- **size**: size of the imposter detail level
  defines the number of snapshots to take around the equator. Imagine the object being rotated around the vertical axis, then a snapshot taken at regularly spaced intervals.
  defines the number of snapshots taken between the poles (top and bottom), at each
polarSteps
equator step. eg. At each equator snapshot, snapshots are taken at regular intervals between the poles.

the detail level to use when generating the snapshots. Note that this is an array index rather than a detail size. So if an object has detail sizes of: 200, 150, and 40, then setting dl to 1 will generate the snapshots using detail size 150.

dl
defines the size of the imposter images in pixels. The larger the number, the more detailed the billboard will be.

dim
flag indicating whether to include the "pole" snapshots. ie. the views from the top and bottom of the object.

includePoles
if pole snapshots are active (includePoles is true), this parameter defines the camera angle (in degrees) within which to render the pole snapshot. eg. if polar_angle is set to 25 degrees, then the snapshot taken at the pole (looking directly down or up at the object) will be rendered when the camera is within 25 degrees of the pole.

polar_angle

Returns:
true if successful, false otherwise

Example:

%this.addImposter( 2, 4, 0, 0, 64, false, %this.addImposter( 2, 4, 2, 0, 64, true, :

bool TSShapeConstructor::addMesh (string meshName, string srcShape, string srcMesh )
Add geometry from another DTS or DAE shape file into this shape.

Any materials required by the source mesh are also copied into this shape.

**Parameters:**

- **meshName**
  full name (object name + detail size) of the new mesh. If no detail size is present at the end of the name, a value of 2 is used.

- **srcShape**
  name of a shape file (DTS or DAE) that contains the mesh.

- **srcMesh**
  the full name (object name + detail size) of the mesh to copy from the DTS/DAE file into this shape.

**Returns:**

true if successful, false otherwise

**Example:**

```plaintext
%this.addMesh(  "ColMesh-1",  "./collision.dts"
%this.addMesh(  "SimpleShape10",  "./testShape.dae"
```

```plaintext
bool TSShapeConstructor::addNode(string name,
string parentName,
TransformF txfm = TransformF::Identity,
bool isWorld = false
)
```

Add a new node.

**Parameters:**
**name**
name for the new node (must not already exist)

**parentName**
name of an existing node to be the parent of the new node. If empty (""), the new node will be at the root level of the node hierarchy.

**txfm**
(optional) transform string of the form: "pos.x pos.y pos.z rot.x rot.y rot.z rot.angle"

**isworld**
(optional) flag to set the local-to-parent or the global transform. If false, or not specified, the position and orientation are treated as relative to the node's parent.

**Returns:**
true if successful, false otherwise

**Example:**

```plaintext
%this.addNode( "Nose", "Bip01 Head", "0 2
%this.addNode( "myRoot", "", "0 0 4 0 0 1
%this.addNode( "Nodes", "Bip01 Head", "0 2
```

```cpp
bool TSShapeConstructor::addPrimitive(string meshName, string type, string params, TransformF txfm, string nodeName)
```

Add a new mesh primitive to the shape.

**Parameters:**

- **meshName**
  full name (object name + detail size) of the new mesh. If no detail size is present at the end of the name, a value of 2 is used.

- **meshName**
  An underscore before the number at the end of the name will be interpreted as a negative sign. eg. "MyMesh_4" will be interpreted as "MyMesh-"
4".

type one of: "box", "sphere", "capsule"

mesh primitive parameters:
  • for box: "size_x size_y size_z"
  • for sphere: "radius"
  • for capsule: "height radius"

params

txfm local transform offset from the node for this mesh

nodeName name of the node to attach the new mesh to (will
change the object's node if adding a new mesh
to an existing object)

Returns:
true if successful, false otherwise

Example:

```plaintext
%this.addMesh( "Box4", "box", "2 4 2", "0
%this.addMesh( "Sphere256", "sphere", "2",
%this.addMesh( "MyCapsule-1", "capsule", '

bool TSShapeConstructor::addSequence( string source,
    string name,
    int start = 0,
    int end = -1,
    bool padRot = true,
    bool padTrans = false
)

Add a new sequence to the shape.

Parameters:

source the name of an existing sequence, or the name of
a DTS or DAE shape or DSQ sequence file. When
the shape file contains more than one sequence,
the desired sequence can be specified by
appending the name to the end of the shape file.
eg. "myShape.dts run" would select the "run" sequence from the "myShape.dts" file.

<table>
<thead>
<tr>
<th>name</th>
<th>name of the new sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>(optional) first frame to copy. Defaults to 0, the first frame in the sequence.</td>
</tr>
<tr>
<td>end</td>
<td>(optional) last frame to copy. Defaults to -1, the last frame in the sequence.</td>
</tr>
<tr>
<td>padRot</td>
<td>(optional) copy root-pose rotation keys for non-animated nodes. This is useful if the source sequence data has a different root-pose to the target shape, such as if one character was in the T pose, and the other had arms at the side. Normally only nodes that are actually rotated by the source sequence have keyframes added, but setting this flag will also add keyframes for nodes that are not animated, but have a different root-pose rotation to the target shape root pose.</td>
</tr>
<tr>
<td>padTrans</td>
<td>(optional) copy root-pose translation keys for non-animated nodes. This is useful if the source sequence data has a different root-pose to the target shape, such as if one character was in the T pose, and the other had arms at the side. Normally only nodes that are actually moved by the source sequence have keyframes added, but setting this flag will also add keyframes for nodes that are not animated, but have a different root-pose position to the target shape root pose.</td>
</tr>
</tbody>
</table>

**Returns:**

true if successful, false otherwise

**Example:**

```plaintext
%this.addSequence( "./testShape.dts ambient"
%this.addSequence( "./myPlayer.dae run", '
%this.addSequence( "./player_look.dsq", "
%this.addSequence( "walk", "walk_shortA",
%this.addSequence( "walk", "walk_shortB",
```
bool TSShapeConstructor::addTrigger(string name, int keyframe, int state)

Add a new trigger to the sequence.

**Parameters:**
- *name* name of the sequence to modify
- *keyframe* keyframe of the new trigger
- *state* of the new trigger

**Returns:**
- true if successful, false otherwise

**Example:**
```cpp
%this.addTrigger( "walk", 3, 1 );
%this.addTrigger( "walk", 5, -1 );
```

```cpp
void TSShapeConstructor::dumpShape(string filename = "")
```

Dump the shape hierarchy to the console or to a file. Useful for reviewing the result of a series of construction commands.

**Parameters:**
- *filename* Destination filename. If not specified, dump to console.

**Example:**
```cpp
%this.dumpShape(); // dump to console
%this.dumpShape( "./dump.txt" ); // dump to file
```
Get the bounding box for the shape.

**Returns:**

Bounding box "minX minY minZ maxX maxY maxZ"

Get the total number of detail levels in the shape.

**Returns:**

the number of detail levels in the shape

Get the index of the detail level with a given size.

**Parameters:**

size  size of the detail level to lookup

**Returns:**

index of the detail level with the desired size, or -1 if no such detail exists

**Example:**

```cpp
if ( %this.getDetailLevelSize( 32 ) == -1 )
    echo( "Error: This shape does not have..."
```

Get the name of the indexed detail level.
Parameters:

\textit{index} detail level index (valid range is 0 - \texttt{getDetailLevelCount()-1})

Returns:

the detail level name

Example:

```
// print the names of all detail levels in the shape
%count = %this.getDetailLevelCount();
for ( %i = 0; %i < %count; %i++)
    echo( %i SPC %this.getDetailLevelName(%i));
```

\begin{Verbatim}
int TSShapeConstructor::getDetailLevelSize(int index )
\end{Verbatim}

Get the size of the indexed detail level.

Parameters:

\textit{index} detail level index (valid range is 0 - \texttt{getDetailLevelCount()-1})

Returns:

the detail level size

Example:

```
// print the sizes of all detail levels in the shape
%count = %this.getDetailLevelCount();
for ( %i = 0; %i < %count; %i++)
    echo( "Detail" @ %i @ " has size " @ %t
```

\begin{Verbatim}
int TSShapeConstructor::getImposterDetailLevel( )
\end{Verbatim}
Get the index of the imposter (auto-billboard) detail level (if any).

**Returns:**

imposter detail level index, or -1 if the shape does not use imposters.

```cpp
string TSShapeConstructor::getImposterSettings(int index)
```

Get the settings used to generate imposters for the indexed detail level.

**Parameters:**

- `index` index of the detail level to query (does not need to be an imposter detail level)

**Returns:**

string of the form: "valid eqSteps pSteps dl dim poles angle",

where:

- **valid**
  1 if this detail level generates imposters, 0 otherwise

- **eqSteps**
  number of steps around the equator

- **pSteps**
  number of steps between the poles

- **dl**
  index of the detail level used to generate imposters

- **dim**
  size (in pixels) of each imposter image

- **poles**
  1 to include pole images, 0 otherwise

- **angle**
  angle at which to display pole images

**Example:**
// print the imposter detail level setting
%index = %this.getImposterDetailLevel();
if ( %index != -1 )
  echo( "Imposter settings: " @ %this.get

int TSShapeConstructor::getMeshCount(string name )
Get the number of meshes (detail levels) for the specified object.

Parameters:
  name  name of the object to query

Returns:
  the number of meshes for this object.

Example:
  %count = %this.getMeshCount( "SimpleShape"

string TSShapeConstructor::getMeshMaterial(string name )
Get the name of the material attached to a mesh. Note that only
the first material used by the mesh is returned.

Parameters:
  name  full name (object name + detail size) of the mesh to
        query

Returns:
  name of the material attached to the mesh (suitable for use
  with the Material mapTo field)

Example:
string TSShapeConstructor::getMeshName(string name, int index)

Get the name of the indexed mesh (detail level) for the specified object.

**Parameters:**

- *name*  name of the object to query
- *index*  index of the mesh (valid range is 0 - `getMeshCount()` - 1)

**Returns:**

the mesh name.

**Example:**

```cpp
// print the names of all meshes in the shape
%objCount = %this.getObjectCount();
for ( %i = 0; %i < %objCount; %i++ )
{
    %objName = %this.getObjectName( %i );
    %meshCount = %this.getMeshCount( %objName );
    for ( %j = 0; %j < %meshCount; %j++ )
        echo( %this.getMeshName( %objName, %j ) );
}
```

int TSShapeConstructor::getMeshSize(string name, int index)


Get the detail level size of the indexed mesh for the specified object.

**Parameters:**

- **name**  name of the object to query
- **index**  index of the mesh (valid range is 0 - `getMeshCount()-1`)

**Returns:**

the mesh detail level size.

**Example:**

```c++
// print sizes for all detail levels of this object
%objName = "trunk";
%count = %this.getMeshCount( %objName );
for ( %i = 0; %i < %count; %i++ )
  echo( %this.getMeshSize( %objName, %i ) );
```

---

**string TSShapeConstructor::getMeshType (string name )**

Get the display type of the mesh.

**Parameters:**

- **name**  name of the mesh to query

**Returns:**

the string returned is one of:

- **normal**  a normal 3D mesh
- **billboard**  a mesh that always faces the camera
- **billboardzaxis**  a mesh that always faces the camera in the Z-axis
Example:

```php
echo( "Mesh type is " @ %this->getMeshType() );
```

```cpp
int TSShapeConstructor::getNodeChildCount(string name )
```

Get the number of children of this node.

**Parameters:**

`name` name of the node to query.

**Returns:**

the number of child nodes.

**Example:**

```cpp
%count = %this->getNodeChildCount( "Bip01" )
```

```cpp
string TSShapeConstructor::getNodeChildName(string name, int index )
```

Get the name of the indexed child node.

**Parameters:**

`name` name of the parent node to query.

`index` index of the child node (valid range is 0 - `getNodeChildName()`-1).

**Returns:**

the name of the indexed child node.

**Example:**

```cpp
function dumpNode( %shape, %name, %indent
```
{ echo( %indent @ %name );
  %count = %shape.getNodeChildCount( %name );
  for ( %i = 0; %i < %count; %i++ )
    dumpNode( %shape, %shape.getNodeChild(%name) );
}

function dumpShape( %shape )
{
  // recursively dump node hierarchy
  %count = %shape.getNodeCount();
  for ( %i = 0; %i < %count; %i++ )
  {
    // dump top level nodes
    %name = %shape.getNodeName( %i );
    if ( %shape.getNodeParentName( %name ) )
      dumpNode( %shape, %name, "" );
  }
}

int TSShapeConstructor::getNodeCount( )

Get the total number of nodes in the shape.

Returns:
  the number of nodes in the shape.

Example:
  %count = %this.getNodeCount();

int TSShapeConstructor::getNodeIndex(string name )
Get the index of the node.

**Parameters:**

*name* name of the node to lookup.

**Returns:**

the index of the named node, or -1 if no such node exists.

**Example:**

```c
// get the index of Bip01 Pelvis node in the shape
%index = %this.getNodeIndex( "Bip01 Pelvis"
```

---

Get the name of the indexed node.

**Parameters:**

*index* index of the node to lookup (valid range is 0 - getNodeCount()-1).

**Returns:**

the name of the indexed node, or "" if no such node exists.

**Example:**

```c
// print the names of all the nodes in the shape
%count = %this.getNodeCount();
for (%i = 0; %i < %count; %i++)
  echo(%i SP %this.getNodeName(%i));
```

---

Get the number of geometry objects attached to this node.
Parameters:

    name  name of the node to query.

Returns:

    the number of attached objects.

Example:

    int count = this.getNodeObjectCount("Bip01 Head");

```
string TSShapeConstructor::getNodeObjectName(string name, int index)
```

Get the name of the indexed object.

Parameters:

    name  name of the node to query.
    index index of the object (valid range is 0 -
            getNodeObjectCount()-1).

Returns:

    the name of the indexed object.

Example:

    // print the names of all objects attached to the node
    int count = this.getNodeObjectCount("Bip01 Head");
    for (int i = 0; i < count; i++)
        echo(this.getNodeObjectName("Bip01 Head"));

```
string TSShapeConstructor::getNodeParentName(string name )
```

Get the name of the node's parent. If the node has no parent (ie. it
is at the root level), return an empty string.

**Parameters:**

- `name` name of the node to query.

**Returns:**

- the name of the node's parent, or "" if the node is at the root level

**Example:**

```cpp
echo( "Bip01 Pelvis parent = " @ %this.getNodeParentName( "Bip01 Pelvis" ) );
```

**TransformF TSShapeConstructor::getNodeTransform**

Get the base (ie. not animated) transform of a node.

**Parameters:**

- `name` name of the node to query.
- `isWorld` true to get the global transform, false (or omitted) to get the local-to-parent transform.

**Returns:**

- the node transform in the form "pos.x pos.y pos.z rot.x rot.y rot.z rot.angle".

**Example:**

```cpp
%ret = %this.getNodeTransform( "mount0" );
%this.setNodeTransform( "mount4", %ret );
```

**int TSShapeConstructor::getObjectCount**
Get the total number of objects in the shape.

**Returns:**
the number of objects in the shape.

**Example:**

```cpp
%count = %this.getObjectCount();
```

---

```cpp
int TSShapeConstructor::getObjectIndex(string name )
```

Get the index of the first object with the given name.

**Parameters:**
- `name` name of the object to get.

**Returns:**
the index of the named object.

**Example:**

```cpp
%index = %this.getObjectIndex( "Head" );
```

---

```cpp
string TSShapeConstructor::getObjectName(int index )
```

Get the name of the indexed object.

**Parameters:**
- `index` index of the object to get (valid range is 0 - `getObjectCount()`-1).

**Returns:**
the name of the indexed object.
Example:
```c++
// print the names of all objects in the shape
%count = %this.getObjectCount();
for ( %i = 0; %i < %count; %i++ )
    echo( %i SPC %this.getObjectName( %i )
```

string TSShapeConstructor::getObjectNode(string name )

Get the name of the node this object is attached to.

**Parameters:**

- *name* name of the object to get.

**Returns:**

the name of the attached node, or an empty string if this object is not attached to a node (usually the case for skinned meshes).

Example:
```c++
echo( "Hand is attached to " @ %this.getObjectNode( "Hand" )
```

string TSShapeConstructor::getSequenceBlend(string name )

Get information about blended sequences.

**Parameters:**

- *name* name of the sequence to query

**Returns:**

TAB delimited string of the form: "isBlend blendSeq blendFrame", where:

- **blend_flag**
a boolean flag indicating whether this sequence is a blend

**blend_seq_name**

the name of the sequence that contains the reference frame (empty for blend sequences embedded in DTS files)

**blend_seq_frame**

the blend reference frame (empty for blend sequences embedded in DTS files)

**Note:**

Note that only sequences set to be blends using the `setSequenceBlend` command will contain the blendSeq and blendFrame information.

**Example:**

```plaintext
%blendData = %this.getSequenceBlend( "look" if ( getField( %blendData, 0 ) )
    echo( "look is a blend, reference: " @
```

---

**int TSShapeConstructor::getSequenceCount( )**

Get the total number of sequences in the shape.

**Returns:**

the number of sequences in the shape

---

**bool TSShapeConstructor::getSequenceCyclic(string name )**

Check if this sequence is cyclic (looping).

**Parameters:**

name name of the sequence to query
Returns:
true if this sequence is cyclic, false if not

Example:
```cpp
if (!%this.getSequenceCyclic( "ambient" ))
    error( "ambient sequence is not cyclic!"
```

```cpp
int TSShapeConstructor::getSequenceFrameCount(string name)
```

Get the number of keyframes in the sequence.

Parameters:
- `name` name of the sequence to query

Returns:
number of keyframes in the sequence

Example:
```cpp
echo( "Run has " @ %this.getSequenceFrameCount( "ambient"
```

```cpp
string TSShapeConstructor::getSequenceGroundSpeed(string name)
```

Get the ground speed of the sequence.

Note:
Note that only the first 2 ground frames of the sequence are examined; the speed is assumed to be constant throughout the sequence.

Parameters:
- `name` name of the sequence to query

Returns:
string of the form: "trans.x trans.y trans.z rot.x rot.y rot.z"

Example:

```cpp
%speed = VectorLen( getWords( %this.getSequenceGroundSpeed() ) )

String echo( "Run moves at " @ %speed @ " units per frame"
```

```cpp
int TSShapeConstructor::getSequenceIndex( string name )

Find the index of the sequence with the given name.

Parameters:

- `name`  name of the sequence to lookup

Returns:

index of the sequence with matching name, or -1 if not found

Example:

```cpp
if ( %this.getSequenceIndex( "walk" ) == -1 )
    echo( "Could not find 'walk' sequence"
```

```cpp
string TSShapeConstructor::getSequenceName( int index )

Get the name of the indexed sequence.

Parameters:

- `index`  index of the sequence to query (valid range is 0 - getSequenceCount()-1)

Returns:

the name of the sequence

Example:
```
// print the name of all sequences in the shape
%count = %this.getSequenceCount();
for ( %i = 0; %i < %count; %i++ )
    echo( %i SPC %this.getSequenceName( %i ) );

float TSShapeConstructor::getSequencePriority(string name )

Get the priority setting of the sequence.

Parameters:

    name name of the sequence to query

Returns:

    priority value of the sequence

string TSShapeConstructor::getSequenceSource(string name )

Get information about where the sequence data came from.

For example, whether it was loaded from an external DSQ file.

Parameters:

    name name of the sequence to query

Returns:

    TAB delimited string of the form: "from reserved start end total", where:
    from
        the source of the animation data, such as the path to a DSQ file, or the name of an existing sequence in the shape. This field will be empty for sequences already embedded in the DTS or DAE file.
    reserved
reserved value

**start**
the first frame in the source sequence used to create this sequence

**end**
the last frame in the source sequence used to create this sequence

**total**
the total number of frames in the source sequence

**Example:**

```javascript
// print the source for the walk animation
echo( "walk source:" SPC getField( %this.
```

```c
int TSShapeConstructor::getTargetCount( )
```

Get the number of materials in the shape.

**Returns:**
the number of materials in the shape.

**Example:**

```c
%count = %this.getTargetCount();
```

```c
string TSShapeConstructor::getTargetName(int index )
```

Get the name of the indexed shape material.

**Parameters:**

`index`  index of the material to get (valid range is 0 -
`getTargetCount()-1`.

**Returns:**
the name of the indexed material.

Example:

```cpp
%count = %this.getTargetCount();
for ( %i = 0; %i < %count; %i++ )
    echo( "Target " @ %i @ ": " @ %this.get
```

Get information about the indexed trigger.

**Parameters:**

- `name` name of the sequence to query
- `index` index of the trigger (valid range is 0 - `getTriggerCount()-1`)

**Returns:**

string of the form "frame state"

**Example:**

```cpp
// print all triggers in the sequence
%count = %this.getTriggerCount( "back" );
for ( %i = 0; %i < %count; %i++ )
    echo( %i SPC %this.getTrigger( "back",
```

Get the number of triggers in the specified sequence.

**Parameters:**
name name of the sequence to query

**Returns:**
number of triggers in the sequence

```cpp
void TSShapeConstructor::notifyShapeChanged()
```

Notify game objects that this shape file has changed, allowing them to update internal data if needed.

```cpp
void TSShapeConstructor::onLoad()
```

Called immediately after the DTS or DAE file has been loaded; before the shape data is available to any other object (StaticShape, Player etc). This is where you should put any post-load commands to modify the shape in-memory such as addNode, renameSequence etc.

```cpp
void TSShapeConstructor::onUnload()
```

Called when the DTS or DAE resource is flushed from memory. Not normally required, but may be useful to perform cleanup.

```cpp
bool TSShapeConstructor::removeDetailLevel(int index)
```

Remove the detail level (including all meshes in the detail level).

**Parameters:**
- size size of the detail level to remove

**Returns:**
- true if successful, false otherwise
Example:

```cpp
%this.removeDetailLevel( 2 );
```

```cpp
bool TSShapeConstructor::removeImposter()
```

Remove the imposter detail level (if any) from the shape.

**Returns:**
true if successful, false otherwise

```cpp
bool TSShapeConstructor::removeMesh(string name)
```

Remove a mesh from the shape.

If all geometry is removed from an object, the object is also removed.

**Parameters:**
- `name` full name (object name + detail size) of the mesh to remove

**Returns:**
true if successful, false otherwise

Example:

```cpp
%this.removeMesh( "SimpleShape128" );
```

```cpp
bool TSShapeConstructor::removeNode(string name)
```

Remove a node from the shape.

The named node is removed from the shape, including from any
sequences that use the node. Child nodes and objects attached to the node are re-assigned to the node's parent.

**Parameters:**

*name* name of the node to remove.

**Returns:**

true if successful, false otherwise.

**Example:**

```cpp
%this.removeNode( "Nose" );
```

bool TSShapeConstructor::removeObject(string name)

Remove an object (including all meshes for that object) from the shape.

**Parameters:**

*name* name of the object to remove.

**Returns:**

true if successful, false otherwise.

**Example:**

```cpp
// clear all objects in the shape
%count = %this.getObjectCount();
for ( %i = %count-1; %i >= 0; %i-- )
    %this.removeObject( %this.getObjectName
```

bool TSShapeConstructor::removeSequence(string name)

Remove the sequence from the shape.
Parameters:

- *name* name of the sequence to remove

Returns:

true if successful, false otherwise

```cpp
bool TSShapeConstructor::removeTrigger (string name,
                                         int   keyframe,
                                         int   state)
```

Remove a trigger from the sequence.

Parameters:

- *name* name of the sequence to modify
- *keyframe* keyframe of the trigger to remove
- *state* of the trigger to remove

Returns:

true if successful, false otherwise

Example:

```cpp
%this.removeTrigger( "walk", 3, 1 );
```

```cpp
bool TSShapeConstructor::renameDetailLevel(string oldName,
                                          string newName)
```

Rename a detail level.

Note:

Note that detail level names must be unique, so this command will fail if there is already a detail level with the
desired name

Parameters:

oldName  current name of the detail level
newName  new name of the detail level

Returns:

true if successful, false otherwise

Example:

%this.renameDetailLevel( "detail-1", "collision-1"

bool TSShapeConstructor::renameNode(string oldName, string newName)

Rename a node.

Note:

Note that node names must be unique, so this command will fail if there is already a node with the desired name

Parameters:

oldName  current name of the node
newName  new name of the node

Returns:

true if successful, false otherwise

Example:

%this.renameNode( "Bip01 L Hand", "mount5"

bool TSShapeConstructor::renameObject(string oldName,
**renameObject**

```cpp
string newName
)
```

Rename an object.

**Note:**
Note that object names must be unique, so this command will fail if there is already an object with the desired name.

**Parameters:**
- `oldName` current name of the object
- `newName` new name of the object

**Returns:**
true if successful, false otherwise

**Example:**
```cpp
%this.renameObject( "MyBox", "Box" );
```

**renameSequence**

```cpp
bool TSShapeConstructor::renameSequence(string oldName, string newName
)
```

Rename a sequence.

**Note:**
Note that sequence names must be unique, so this command will fail if there is already a sequence with the desired name.

**Parameters:**
- `oldName` current name of the sequence
- `newName` new name of the sequence

**Returns:**
true if successful, false otherwise

Example:

%this.renameSequence( "walking", "walk" );

void TSShapeConstructor::saveShape(string filename )

Save the shape (with all current changes) to a new DTS file.

Parameters:

filename Destination filename.

Example:

%this.saveShape( "/myShape.dts" );

bool TSShapeConstructor::setBounds(Box3F bbox )

Set the shape bounds to the given bounding box.

Parameters:

Bounding box "minX minY minZ maxX maxY maxZ"

Returns:

true if successful, false otherwise

int TSShapeConstructor::setDetailLevelSize(int index, int newSize )

Change the size of a detail level.

Note:
Note that detail levels are always sorted in decreasing size order, so this command may cause detail level indices to change.

**Parameters:**

- `index` index of the detail level to modify
- `newSize` new size for the detail level

**Returns:**

new index for this detail level

**Example:**

```
%this.setDetailLevelSize( 2, 256 );
```

```cpp
bool TSShapeConstructor::setMeshMaterial(string meshName, string matName)
```

Set the name of the material attached to the mesh.

**Parameters:**

- `meshName` full name (object name + detail size) of the mesh to modify
- `matName` name of the material to attach. This could be the base name of the diffuse texture (eg. "test_mat" for "test_mat.jpg"), or the name of a Material object already defined in script.

**Returns:**

true if successful, false otherwise

**Example:**

```
// set the mesh material
%this.setMeshMaterial( "SimpleShape128", ''
```
bool TSShapeConstructor::setMeshSize(string name,
int size)

Change the detail level size of the named mesh.

**Parameters:**

- `name` full name (object name + current size) of the mesh to modify
- `size` new detail level size

**Returns:**

true if successful, false otherwise.

**Example:**

```cpp
%this.setMeshSize( "SimpleShape128", 64 );
```

---

bool TSShapeConstructor::setMeshType(string name,
string type)

Set the display type for the mesh.

**Parameters:**

- `name` full name (object name + detail size) of the mesh to modify
- `type` the new type for the mesh: "normal", "billboard" or "billboardzaxis"

**Returns:**

true if successful, false otherwise

**Example:**

---

```cpp
```
// set the mesh to be a billboard
%this.setMeshType( "SimpleShape64", "billboard"

bool TSShapeConstructor::setNodeParent(string name, string parentName)

Set the parent of a node.

**Parameters:**
- `name`  
  name of the node to modify
- `parentName`  
  name of the parent node to set (use "" to move the node to the root level)

**Returns:**
true if successful, false if failed

**Example:**
%this.setNodeParent( "Bip01 Pelvis", "start01"

bool TSShapeConstructor::setNodeTransform(string name, TransformF txfm, bool isWorld = f)

Set the base transform of a node. That is, the transform of the node when in the root (not-animated) pose.

**Parameters:**
- `name`  
  name of the node to modify
- `txfm`  
  transform string of the form: "pos.x pos.y pos.z rot.x rot.y rot.z rot.angle"  
  (optional) flag to set the local-to-parent or the global
isworld transform. If false, or not specified, the position and orientation are treated as relative to the node's parent.

**Returns:**
true if successful, false otherwise

**Example:**

```%this.setNodeTransform( "mount0", "0 0 1 0"
%this.setNodeTransform( "mount0", "0 0 0 0"
%this.setNodeTransform( "mount0", "1 0 0 0"
```

---

```bool TSShapeConstructor::setObjectNode (string objName, string nodeName)
```

Set the node an object is attached to.

When the shape is rendered, the object geometry is rendered at the node's current transform.

**Parameters:**

- `objName` name of the object to modify
- `nodeName` name of the node to attach the object to

**Returns:**
true if successful, false otherwise

**Example:**

```%this.setObjectNode( "Hand", "Bip01 LeftHand"
```
Mark a sequence as a blend or non-blend.

A blend sequence is one that will be added on top of any other playing sequences. This is done by storing the animated node transforms relative to a reference frame, rather than as absolute transforms.

**Parameters:**

- **name** name of the sequence to modify
- **blend** true to make the sequence a blend, false for a non-blend
- **blendSeq** the name of the sequence that contains the blend reference frame
- **blendFrame** the reference frame in the blendSeq sequence

**Returns:**

true if successful, false otherwise

**Example:**

```c++
%this.setSequenceBlend( "look", true, "root"
```

```c++
bool TSShapeConstructor::setSequenceCyclic(string name,
                                          bool cyclic )
```

Mark a sequence as cyclic or non-cyclic.

**Parameters:**

- **name** name of the sequence to modify
- **cyclic** true to make the sequence cyclic, false for non-cyclic
Returns:
true if successful, false otherwise

Example:

```cpp
%this.setSequenceCyclic( "ambient", true );
%this.setSequenceCyclic( "shoot", false );
```

```cpp
bool TSShapeConstructor::setSequenceGroundSpeed( string name, Point3F transSpeed, Point3F rotSpeed )
```

Set the translation and rotation ground speed of the sequence.

The ground speed of the sequence is set by generating ground transform keyframes. The ground translational and rotational speed is assumed to be constant for the duration of the sequence. Existing ground frames for the sequence (if any) will be replaced.

**Parameters:**

- `name` name of the sequence to modify
- `transSpeed` translational speed (trans.x trans.y trans.z) in Torque units per frame
- `rotSpeed` (optional) rotational speed (rot.x rot.y rot.z) in radians per frame. Default is "0 0 0"

**Returns:**
true if successful, false otherwise

**Example:**

```cpp
%this.setSequenceGroundSpeed( "run", "5 0 0" );
%this.setSequenceGroundSpeed( "spin", "0 0 0" );
```
bool TSShapeConstructor::setSequencePriority(string name, float priority)

Set the sequence priority.

**Parameters:**
- *name* name of the sequence to modify
- *priority* new priority value

**Returns:**
- true if successful, false otherwise

void TSShapeConstructor::writeChangeSet()

Write the current change set to a TSShapeConstructor script file. The name of the script file is the same as the model, but with .cs extension. eg. myShape.cs for myShape.dts or myShape.dae.
Member Data Documentation

bool TSShapeConstructor::adjustCenter

Translate COLLADA model on import so the origin is at the center. No effect for DTS files.

bool TSShapeConstructor::adjustFloor

Translate COLLADA model on import so origin is at the (Z axis) bottom of the model. No effect for DTS files.

This can be used along with adjustCenter to have the origin at the center of the bottom of the model.

See also:
   adjustCenter

string TSShapeConstructor::alwaysImport

TAB separated patterns of nodes to import even if in neverImport list. No effect for DTS files.

Torque allows unwanted nodes in COLLADA (.dae) files to to be ignored during import. This field contains a TAB separated list of patterns to match node names. Any node that matches one of the patterns in the list will always be imported, even if it also matches the neverImport list

See also:
   neverImport

Example:

singleton TSShapeConstructor(MyShapeDae)
string TSShapeConstructor::alwaysImportMesh

TAB separated patterns of meshes to import even if in neverImportMesh list. No effect for DTS files.

Torque allows unwanted meshes in COLLADA (.dae) files to be ignored during import. This field contains a TAB separated list of patterns to match mesh names. Any mesh that matches one of the patterns in the list will **always** be imported, even if it also matches the neverImportMesh list

**See also:**
neverImportMesh

**Example:**

```cpp
singleton TSShapeConstructor(MyShapeDae)
{
    baseShape = "./myShape.dae"
    alwaysImportMesh = "body*" TAB "armor"
    neverImportMesh = "*-dummy"
}
```

**filename TSShapeConstructor::baseShape**

Specifies the path to the DTS or DAE file to be operated on by this object.
Since the **TSShapeConstructor** script must be in the same folder as the DTS or DAE file, it is recommended to use a relative path so that the shape and script files can be copied to another location without having to modify the path.

```
bool TSShapeConstructor::forceUpdateMaterials
```

Forces update of the materials.cs file in the same folder as the COLLADA (.dae) file, even if Materials already exist. No effect for DTS files.

Normally only Materials that are not already defined are written to materials.cs.

```
bool TSShapeConstructor::ignoreNodeScale
```

Ignore `<scale>` elements inside COLLADA `<node>`s. No effect for DTS files.

This field is a workaround for certain exporters that generate bad node scaling, and is not usually required.

```
TSShapeConstructorLodType TSShapeConstructor::lodType
```

Control how the COLLADA (.dae) importer interprets LOD in the model. No effect for DTS files.

Set to one of the following values:

**DetectDTS**

The default value. Instructs the importer to search for a 'baseXXX->startXXX' node hierarchy at the root level. If found, the importer acts as if "TrailingNumber" was set. Otherwise, all geometry is imported at a single detail size.

**SingleSize**
All geometry is imported at a fixed detail size. Numbers at the end of geometry node's are ignored.

**TrailingNumber**

Numbers at the end of geometry node's name are interpreted as the detail size (similar to DTS exporting). Geometry instances with the same base name but different trailing number are grouped into the same object.

**DEFAULT**

The default value. Use the value in the .dae file (defaults to Z_AXIS if the <up_axis> element is not present).

---

**string TSShapeConstructor::matNamePrefix**

Prefix to apply to all material map names in the COLLADA (.dae) file. No effect for DTS files.

This field is useful to avoid material name clashes for exporters that generate generic material names like "texture0" or "material1".

---

**string TSShapeConstructor::neverImport**

TAB separated patterns of nodes to ignore on loading. No effect for DTS files.

Torque allows unwanted nodes in COLLADA (.dae) files to to be ignored during import. This field contains a TAB separated list of patterns to match node names. Any node that matches one of the patterns in the list will not be imported (unless it matches the alwaysImport list.

**See also:**

alwaysImport

---

**string TSShapeConstructor::neverImportMesh**
TAB separated patterns of meshes to ignore on loading. No effect for DTS files.

Torque allows unwanted meshes in COLLADA (.dae) files to be ignored during import. This field contains a TAB separated list of patterns to match mesh names. Any mesh that matches one of the patterns in the list will not be imported (unless it matches the alwaysImportMesh list).

**See also:**
- alwaysImportMesh

<table>
<thead>
<tr>
<th>filename TSShapeConstructor::sequence</th>
</tr>
</thead>
</table>

Legacy method of adding sequences to a DTS or DAE shape after loading.

**Example:**

```cpp
singleton TSShapeConstructor(MyShapeDae) {
    baseShape = ".//myShape.dae";
    sequence = "../anims/root.dae root";
    sequence = "../anims/walk.dae walk";
    sequence = "../anims/jump.dsq jump";
}
```

<table>
<thead>
<tr>
<th>int TSShapeConstructor::singleDetailSize</th>
</tr>
</thead>
</table>

Sets the detail size when lodType is set to SingleSize. No effect otherwise, and no effect for DTS files.

**See also:**
- lodType
**float TSShapeConstructor::unit**

Override the `<unit>` element in the COLLADA (.dae) file. No effect for DTS files.

COLLADA (.dae) files usually contain a `<unit>` element that indicates the 'real world' units that the model is described in. It means you can work in sensible and meaningful units in your modeling app.

For example, if you were modeling a small object like a cup, it might make sense to work in inches (1 MAX unit = 1 inch), but if you were modeling a building, it might make more sense to work in feet (1 MAX unit = 1 foot). If you export both models to COLLADA, T3D will automatically scale them appropriately: 1 T3D unit = 1 meter, so the cup would be scaled down by 0.0254, and the building scaled down by 0.3048, given them both the correct scale relative to each other.

Omit the field or set to -1 to use the value in the .dae file (1.0 if the `<unit>` element is not present)

**TSShapeConstructorUpAxis TSShapeConstructor::upAxis**

Override the `<up_axis>` element in the COLLADA (.dae) file. No effect for DTS files.

Set to one of the following values:

**X_AXIS**

Positive X points up. Model will be rotated into Torque's coordinate system (Z up).

**Y_AXIS**

Positive Y points up. Model will be rotated into Torque's coordinate system (Z up).

**Z_AXIS**

Positive Z points up. No rotation will be applied to the model.

**DEFAULT**
The default value. Use the value in the .dae file (defaults to Z_AXIS if the <up_axis> element is not present).

Copyright © GarageGames, LLC. All Rights Reserved.
TSStatic Class Reference
[Game Objects]

A static object derived from a 3D model file and placed within the game world. More...

Inheritance diagram for TSStatic:

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td><code>changeMaterial</code></td>
<td>(string mapTo=&quot;&quot;, Material oldMat=NULL, Material newMat=NULL) Change one of the materials on the shape.</td>
</tr>
<tr>
<td>string</td>
<td><code>getModelFile</code></td>
<td>() Get the model filename used by this shape.</td>
</tr>
<tr>
<td>int</td>
<td><code>getTargetCount</code></td>
<td>() Get the number of materials in the shape.</td>
</tr>
<tr>
<td>string</td>
<td><code>getTargetName</code></td>
<td>(int index=0) Get the name of the indexed shape material.</td>
</tr>
</tbody>
</table>
## Public Attributes

### Collision

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>allowPlayerStep</td>
<td>Allow a Player to walk up sloping polygons in the TSStatic (based on the collisionType).</td>
</tr>
<tr>
<td>TSMatchType</td>
<td>collisionType</td>
<td>The type of mesh data to use for collision queries.</td>
</tr>
<tr>
<td>TSMatchType</td>
<td>decalType</td>
<td>The type of mesh data used to clip decal polygons against.</td>
</tr>
</tbody>
</table>

### Debug

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>forceDetail</td>
<td>Forces rendering to a particular detail level.</td>
</tr>
<tr>
<td>float</td>
<td>renderNormals</td>
<td>Debug rendering mode shows the normals for each point in the TSStatic's mesh.</td>
</tr>
</tbody>
</table>

### Rendering

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>meshCulling</td>
<td>Enables detailed culling of meshes within the TSStatic. Should only be used with large complex shapes like buildings which contain many submeshes.</td>
</tr>
<tr>
<td>bool</td>
<td>originSort</td>
<td>Enables translucent sorting of the TSStatic by its origin instead of the bounds.</td>
</tr>
<tr>
<td>bool</td>
<td>playAmbient</td>
<td>Enables automatic playing of the animation</td>
</tr>
</tbody>
</table>
sequence named "ambient" (if it exists) when the **TSStatic** is loaded.

### Media

<table>
<thead>
<tr>
<th>filename</th>
<th>shapeName</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path and filename of the model file (.DTS, .DAE) to use for this <strong>TSStatic</strong>.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>skin</th>
</tr>
</thead>
<tbody>
<tr>
<td>The skin applied to the shape.</td>
<td></td>
</tr>
</tbody>
</table>
Static Public Attributes

<table>
<thead>
<tr>
<th>static bool</th>
<th>isRenderable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static bool</th>
<th>isSelectable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
**Detailed Description**

A static object derived from a 3D model file and placed within the game world.

**TSStatic** is the most basic 3D shape in Torque. Unlike **StaticShape** it doesn't make use of a datablock. It derives directly from **SceneObject**. This makes **TSStatic** extremely light weight, which is why the Tools use this class when you want to drop in a DTS or DAE object.

While a **TSStatic** doesn't provide any motion -- it stays were you initially put it -- it does allow for a single ambient animation sequence to play when the object is first added to the scene.

**Example:**

```cpp
new TSStatic(Team1Base) {
    shapeName = "art/shapes/desertStructures/station01.dts"
    playAmbient = "1";
    receiveSunLight = "1";
    receiveLMLighting = "1";
    useCustomAmbientLighting = "0";
    customAmbientLighting = "0 0 0 1";
    collisionType = "Visible Mesh";
    decalType = "Collision Mesh";
    allowPlayerStep = "1";
    renderNormals = "0";
    forceDetail = "-1";
    position = "315.18 -180.418 244.313";
    rotation = "0 0 1 195.952";
    scale = "1 1 1";
    isRenderEnabled = "true";
    canSaveDynamicFields = "1";
};
```
Member Function Documentation

void TSStatic::changeMaterial (string mapTo = "", Material oldMat = NULL, Material newMat = NULL)

Change one of the materials on the shape.

This method changes materials per mapTo with others. The material that is being replaced is mapped to unmapped_mat as a part of this transition.

Note:
Warning, right now this only sort of works. It doesn't do a live update like it should.

Parameters:
- **mapTo** the name of the material target to remap (from getTargetName)
- **oldMat** the old Material that was mapped
- **newMat** the new Material to map

Example:

```
// remap the first material in the shape
%mapTo = %obj.getTargetName( 0 );
%obj.changeMaterial( %mapTo, 0, MyMaterial ]
```

string TSStatic::getModelFile( )

Get the model filename used by this shape.

Returns:
- the shape filename
Example:

```cpp
// Acquire the model filename used on this
modelFilename = %obj.getModelFile();
```

```cpp
int TSStatic::getTargetCount( )
```

Get the number of materials in the shape.

**Returns:**

the number of materials in the shape.

**See also:**

`getTargetName()`

```cpp
string TSStatic::getTargetName(int index = 0 )
```

Get the name of the indexed shape material.

**Parameters:**

\[ index \] index of the material to get (valid range is 0 - `getTargetCount()-1`).

**Returns:**

the name of the indexed material.

**See also:**

`getTargetCount()`
# Member Data Documentation

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bool TSStatic::allowPlayerStep</code></td>
<td>Allows a <strong>Player</strong> to walk up sloping polygons in the <strong>TSStatic</strong> (based on the <code>collisionType</code>). When set to false, the slightest bump will stop the player from walking on top of the object.</td>
</tr>
<tr>
<td><code>TSMeshType TSStatic::collisionType</code></td>
<td>The type of mesh data to use for collision queries.</td>
</tr>
<tr>
<td><code>TSMeshType TSStatic::decalType</code></td>
<td>The type of mesh data used to clip decal polygons against.</td>
</tr>
<tr>
<td><code>int TSStatic::forceDetail</code></td>
<td>Forces rendering to a particular detail level.</td>
</tr>
<tr>
<td><code>bool TSStatic::meshCulling</code></td>
<td>Enables detailed culling of meshes within the <strong>TSStatic</strong>. Should only be used with large complex shapes like buildings which contain many submeshes.</td>
</tr>
<tr>
<td><code>bool TSStatic::originSort</code></td>
<td></td>
</tr>
</tbody>
</table>
Enables translucent sorting of the `TSStatic` by its origin instead of the bounds.

```cpp
bool TSStatic::playAmbient
```

Enables automatic playing of the animation sequence named "ambient" (if it exists) when the `TSStatic` is loaded.

```cpp
float TSStatic::renderNormals
```

Debug rendering mode shows the normals for each point in the `TSStatic`'s mesh.

```cpp
filename TSStatic::shapeName
```

Path and filename of the model file (.DTS, .DAE) to use for this `TSStatic`.

```cpp
string TSStatic::skin
```

The skin applied to the shape.

'Skinning' the shape effectively renames the material targets, allowing different materials to be used on different instances of the same model.

Any material targets that start with the old skin name have that part of the name replaced with the new skin name. The initial old skin name is "base". For example, if a new skin of "blue" was applied to a model that had material targets `base_body` and `face`, the new targets would be `blue_body` and `face`. Note that `face` was not renamed since it did not start with the old skin name of "base".
To support models that do not use the default "base" naming convention, you can also specify the part of the name to replace in the skin field itself. For example, if a model had a material target called `shapemat`, we could apply a new skin "shape=blue", and the material target would be renamed to `bluemat` (note "shape" has been replaced with "blue").

Multiple skin updates can also be applied at the same time by separating them with a semicolon. For example: "base=blue;face=happy_face".

Material targets are only renamed if an existing Material maps to that name, or if there is a diffuse texture in the model folder with the same name as the new target.
TurretShape Class Reference
[Game Objects]

Base turret class. More...

Inheritance diagram for TurretShape:

```
  +-----------------+  +-----------------+
  | SimObject       |  | NetObject        |
  +-----------------+  +-----------------+
  |                 |  | SceneObject      |
  +-----------------+  +-----------------+
  |                 |  | GameBase         |
  +-----------------+  +-----------------+
  |                 |  | ShapeBase        |
  +-----------------+  +-----------------+
  | Item            |  | TurretShape      |
  +-----------------+  +-----------------+
  | AllTurretShape  |  |                 |
```

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>doRespawn()</code></td>
<td>Does the turret respawn after it has been destroyed.</td>
</tr>
<tr>
<td>bool</td>
<td><code>getAllowManualFire()</code></td>
<td>Get if the turret is allowed to fire through moves.</td>
</tr>
<tr>
<td>bool</td>
<td><code>getAllowManualRotation()</code></td>
<td>Get if the turret is allowed to rotate through moves.</td>
</tr>
<tr>
<td>string</td>
<td><code>getState()</code></td>
<td>Get the name of the turret's current state.</td>
</tr>
<tr>
<td>Point3F</td>
<td><code>getTurretEulerRotation()</code></td>
<td>Get Euler rotation of this turret's heading and pitch nodes.</td>
</tr>
<tr>
<td>void</td>
<td><code>setAllowManualFire</code>(bool allow)</td>
<td>Set if the turret is allowed to fire through moves.</td>
</tr>
<tr>
<td>void</td>
<td><code>setAllowManualRotation</code>(bool allow)</td>
<td>Set if the turret is allowed to rotate through moves.</td>
</tr>
<tr>
<td>void</td>
<td><code>setTurretEulerRotation</code>(Point3F rot)</td>
<td>Set Euler rotation of this turret's heading and pitch nodes in degrees.</td>
</tr>
</tbody>
</table>
### Public Attributes

<table>
<thead>
<tr>
<th>bool</th>
<th>respawn</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Respawn the turret after it has been destroyed.</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>static bool</code></td>
<td><strong>isRenderable</strong></td>
</tr>
<tr>
<td></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>static bool</code></td>
<td><strong>isSelectable</strong></td>
</tr>
<tr>
<td></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Base turret class.

Uses the TurretShapeData datablock for common properties.

The TurretShape class provides a player mountable turret. It also forms the base forAITurretShape, an AI controlled turret. It is based on the Item class, which allows turrets to be treated as any other Item by the Player, such as throwing smaller turrets. When used directly, TurretShape takes input moves from the player's GameConnection to rotate the turret and trigger its weapons.

A turret consists of two components. There is the TurretShape object (or AITurretShape), and then there are one or more ShapeBaseImageData objects that are mounted to the turret. The TurretShape provides the weapons platform that rotates towards a target. The ShapeBaseImageData provides the actual weapon that fires at the target. Any standard ShapeBaseImageData weapon may be used.
Shape File Nodes

The shape file used for the TurretShape needs to have a number of defined nodes. The first node is named 'heading'. The heading node is special in that it is controlled by the TurretShape code. This means that it should not be animated by the artist, nor should it have anything but the default transform applied to it. This doesn't stop the heading node's parent or its children from being animated, however.

The second special node is named 'pitch'. The pitch node is also controlled by the TurretShape code so it too should not be animated within the shape file. Typically the pitch node will be a child of the heading node, although it need not be a direct child. The pitch node is also optional if you don't want the TurretShape to pitch towards its target. In this case you may be doing something special with the mounted weapon to have its projectiles automatically aim towards the target.

The next set of nodes are weaponMount0 through weaponMount3. These provide up to four mounting points for weapons on the turret. Typically these are children of the pitch nodes, although they need not be direct children of that node. You do not need to have all of these weapon mount point nodes defined within the shape. Only as many as you need for the weapons. The mounted ShapeBaseImageData weapons' mountPoint node will mount to these nodes.

There are four optional nodes named pitch0 through pitch3 that may be used in special cases. These nodes are also controlled by the TurretShape code and have the same restrictions. Their rotation exactly matches that of the standard pitch node. These exist for mounted weapons that may not all rotate about the same x axis. For example, a turret may have two sets of weapons, one mounted above the other. These two sets of weapons could all share the same point of rotation (the pitch node) which means they'll rotate as a group. Or the top weapons could be attached to the pitch node while the bottom weapons could be attached to the pitch0 node. This makes the two sets of weapons rotate about their own centers and provides an entirely different look.
You could also use these additional pitchN nodes to animate some non-weapon attachments on the turret, such as a radar dish or targeting scope. TurretShape also supports four optional heading0 through heading3 nodes that operate in the same way as the pitchN nodes.
Weapon Mounting

*TurretShape* weapon mounting is done within the *TurretShapeData::onAdd()* script method. This method makes use of datablock fields that are only defined in script and are not passed along to the client. The first field is *numWeaponMountPoints* that defines the number of weapons that will be mounted and the number of *weaponMountN* nodes to expect within the turret's shape file.

The other fields that are required to mount weapons are the *weapon[]*, *weaponAmmo[]* and *weaponAmmoAmount[]* arrays -- one of each per weapon to mount. The *weapon[]* array points to an *ItemData* datablock that defines the weapon (just like any *Player* weapon). The *weaponAmmo[]* array points to an *ItemData* datablock that defines the ammo to use for the weapon. Finally, the *weaponAmmoAmount[]* array is the quantity of ammo the turret has for that weapon.

As turrets use the same inventory system as players, you also need to define the maximum number of weapons and ammo that the turret may possess. Here is an example of setting up three weapons and their ammo for a turret (a *TurretShapeData* fragment):

**Example:**

```cpp
// Weapon mounting
numWeaponMountPoints = 3;

weapon[0] = TurretWeapon;
weaponAmmo[0] = BulletAmmo;
weaponAmmoAmount[0] = 10000;

weapon[1] = TurretWeaponB;
weaponAmmo[1] = BulletAmmo;
weaponAmmoAmount[1] = 10000;

```
weaponAmmo[2] = BulletAmmo;
weaponAmmoAmount[2] = 10000;

maxInv[TurretWeaponB] = 1;
maxInv[TurretWeapon] = 2;
maxInv[BulletAmmo] = 10000;
Mounted Weapon States

There are a couple of things to be aware of so that an turret's mounted weapons play along with the turret's states, especially for AI turrets. Setting `TurretShapeData::startLoaded` to true indicates that all mounted weapons will start loaded when their state machines start up. A static turret placed with the World Editor would normally begin this way. Setting `TurretShapeData::startLoaded` to false causes all mounted weapons to not start in a loaded state. This can be used to have the mounted weapons begin in some folded state when a deployable turret is thrown by the player. When a thrown turret comes to rest and begins to deploy, all mounted weapons are automatically set to the loaded state so they may also unfold, start up, or show some other method that the weapon is becoming ready to fight. This could also be used for player mountable turrets so that the weapons come to life when a player mounts the turret.

The default scripts for `AITurretShapeData` also fires the first image generic trigger on all mounted weapons when the turret is destroyed. This shows up as `stateTransitionGeneric0In` within a weapon image's state machine. This allows for all weapons to show that they are destroyed or shutting down. Something similar could be done for general `TurretShapeData` turrets.

Weapons can also feed back to the turret they are mounted on. `TurretShape` supports the standard `ShapeBaseImageData` stateRecoil and will play the indicated animation, if available. You can also use `ShapeBaseImageData`'s `stateShapeSequence` field to play a generic sequence on the turret at any time from a mounted weapon.
**Player Mounting**

Turrets act very similar to vehicles when it comes to player mounting. By default colliding with a turret causes the player to mount it, if the turret is free.

When it comes to firing the turret's weapons there are a number of methods that are triggered based on the weaponLinkType on the TurretShapeData datablock. Setting this field to FireTogether causes all weapons to fire at once based on the input from trigger 0. Using GroupedFire will make weaponMount0 and weaponMount2 mounted weapons fire on trigger 0, and weaponMount1 and weaponMount3 mounted weapons fire on trigger 1. Finally, IndividualFire will have each weaponMountN mounted weapons fire based on their own trigger (0 through 3). This provides exact control over which turret weapon will fire when there are multiple weapons mounted.

The player mounting callbacks are done using the TurretBaseData datablock on the server, and in a special case on the TurretBase object on the client. The server side makes use of the standard TurretBaseData::onMountObject() and TurretBaseData::onUnmountObject() callbacks. See those for more information.

When a player mounted turret is destroyed the TurretShapeData::damage() method will automatically kill all mounted players. To modify this behaviour -- such as only dismounting players from a destroyed turret -- you'll need to create your own damage() method for your turret's datablock.

On the client side the special turretMountCallback() callback function is called for the TurretShape object that is being mounted. This callback receives the SimObjectID of the turret object, the SimObjectID of the player doing the mounting or unmounting, and a Boolean set to true if mounting and false if unmounting. As this callback is made on the client, it allows the client to set up any action maps, make HUD changes, etc.

**Example:**
// Turret Support

// Called by the TurretShape class when a player mounts or unmounts it.
// %turret = The turret that was mounted
// %player = The player doing the mounting
// %mounted = True if the turret was mounted

function turretMountCallback(%turret, %player, %mounted)
{
    echo ( "\c4turretMountCallback -> " @ %1, %mounted );

    if (%mounted)
    {
        // Push the action map
        turretMap.push();
    }
    else
    {
        // Pop the action map
        turretMap.pop();
    }
}
Turret Destruction

When a turret is destroyed the default TurretBaseData::onDestroyed() method is called. This causes the turret to sit in a Dead state for TurretBase::DestroyedFadeDelay milliseconds, and then the turret will fade away. If the turret is marked to respawn -- TurretShape::doRespawn() returns true -- then the turret is respawned after TurretShape::RespawnTime milliseconds. By default all turrets placed in the World Editor are marked to respawn.
Turret Optional Animation Sequences

If present in the TurretShape's shape, the optional 'heading' and 'pitch' sequences will be played as the turret rotates. These sequences are given a timeline position that corresponds to the turret's rotation within its minimum and maximum ranges. These sequences could be used to rotate wheels or gears on the turret as it rotates, for example.

See also:
- TurretShapeData
- AITurretShape
- Item
- ShapeBaseImageData
### Member Function Documentation

#### bool TurretShape::doRespawn( )

Does the turret respawn after it has been destroyed.

**Returns:**

- True if the turret respawns.

#### bool TurretShape::getAllowManualFire( )

Get if the turret is allowed to fire through moves.

**Returns:**

- True if the turret is allowed to fire through moves.

#### bool TurretShape::getAllowManualRotation( )

Get if the turret is allowed to rotate through moves.

**Returns:**

- True if the turret is allowed to rotate through moves.

#### string TurretShape::getState( )

Get the name of the turret's current state.

The state is one of the following:

- Dead - The TurretShape is destroyed.
- Mounted - The TurretShape is mounted to an object such as a vehicle.
- Ready - The TurretShape is free to move. The usual state.
**Returns:**
The current state; one of: "Dead", "Mounted", "Ready"

**Point3F TurretShape::getTurretEulerRotation( )**
Get Euler rotation of this turret's heading and pitch nodes.

**Returns:**
the orientation of the turret's heading and pitch nodes in the form of rotations around the X, Y and Z axes in degrees.

**void TurretShape::setAllowManualFire(bool allow )**
Set if the turret is allowed to fire through moves.

**Parameters:**
*allow* If true then the turret may be fired through moves.

**void TurretShape::setAllowManualRotation(bool allow )**
Set if the turret is allowed to rotate through moves.

**Parameters:**
*allow* If true then the turret may be rotated through moves.

**void TurretShape::setTurretEulerRotation(Point3F rot )**
Set Euler rotation of this turret's heading and pitch nodes in degrees.

**Parameters:**
The rotation in degrees. The pitch is the X component and
rot the heading is the Z component. The Y component is ignored.
Member Data Documentation

bool TurretShape::respawn

Respawn the turret after it has been destroyed.

If true, the turret will respawn after it is destroyed.
TurretShapeData Class Reference
[Game Objects]

Defines properties for a TurretShape object. More...

Inheritance diagram for TurretShapeData:

```
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SimObject</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>-------------------</td>
</tr>
<tr>
<td>SimDataBlock</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>-------------------</td>
</tr>
<tr>
<td>GameBaseData</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>-------------------</td>
</tr>
<tr>
<td>ShapeBaseData</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>-------------------</td>
</tr>
<tr>
<td>ItemData</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>-------------------</td>
</tr>
<tr>
<td>TurretShapeData</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>-------------------</td>
</tr>
<tr>
<td>AllTurretShapeData</td>
</tr>
</tbody>
</table>

[legend]
```

List of all members.
Public Member Functions

Callbacks

<table>
<thead>
<tr>
<th>void</th>
<th>onMountObject (TurretShape turret, SceneObject obj, int node)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Informs the TurretShapeData object that a player is mounting it.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>onStickyCollision (TurretShape obj)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Informs the TurretData object that it is now sticking to another object.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>onUnmountObject (TurretShape turret, SceneObject obj)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Informs the TurretShapeData object that a player is unmounting it.</td>
</tr>
</tbody>
</table>
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>cameraOffset</code></td>
<td>Vertical (Z axis) height of the camera above the turret.</td>
</tr>
<tr>
<td>float</td>
<td><code>headingRate</code></td>
<td>Degrees per second rotation.</td>
</tr>
<tr>
<td>float</td>
<td><code>maxHeading</code></td>
<td>Maximum number of degrees to rotate from center.</td>
</tr>
<tr>
<td>float</td>
<td><code>maxPitch</code></td>
<td>Maximum number of degrees to rotate up from straight ahead.</td>
</tr>
<tr>
<td>float</td>
<td><code>minPitch</code></td>
<td>Minimum number of degrees to rotate down from straight ahead.</td>
</tr>
<tr>
<td>float</td>
<td><code>pitchRate</code></td>
<td>Degrees per second rotation.</td>
</tr>
<tr>
<td>bool</td>
<td><code>startLoaded</code></td>
<td>Does the turret's mounted weapon(s) start in a loaded state.</td>
</tr>
<tr>
<td>string</td>
<td><code>TurretShapeFireLinkType</code></td>
<td>weaponLinkType Set how the mounted weapons are linked and triggered.</td>
</tr>
<tr>
<td>bool</td>
<td><code>zRotOnly</code></td>
<td>Should the turret allow only z rotations.</td>
</tr>
</tbody>
</table>
Detailed Description

Defines properties for a TurretShape object.

See also:

TurretShape
TurretShapeData
void TurretShapeData::onMountObject (TurretShape turret, SceneObject obj, int node)

Informs the TurretShapeData object that a player is mounting it.

**Parameters:**
- *turret* The TurretShape object.
- *obj* The player that is mounting.
- *node* The node the player is mounting to.

**Note:**
Server side only.

void TurretShapeData::onStickyCollision (TurretShape obj)

Informs the TurretData object that it is now sticking to another object.

This callback is only called if the TurretData::sticky property for this Turret is true.

**Parameters:**
- *obj* The Turret object that is colliding.

**Note:**
Server side only.

**See also:**
TurretShape, TurretData
**void TurretShapeData::onUnmountObject(TurretShape turret, SceneObject obj)**

Informs the *TurretShapeData* object that a player is unmounting it.

**Parameters:**

- `turret` The *TurretShape* object.
- `obj` The player that is unmounting.

**Note:**

Server side only.
**Member Data Documentation**

<table>
<thead>
<tr>
<th>float TurretShapeData::cameraOffset</th>
<th>Vertical (Z axis) height of the camera above the turret.</th>
</tr>
</thead>
<tbody>
<tr>
<td>float TurretShapeData::headingRate</td>
<td>Degrees per second rotation. A value of 0 means no rotation is allowed. A value less than 0 means the rotation is instantaneous.</td>
</tr>
<tr>
<td>float TurretShapeData::maxHeading</td>
<td>Maximum number of degrees to rotate from center. A value of 180 or more degrees indicates the turret may rotate completely around.</td>
</tr>
<tr>
<td>float TurretShapeData::maxPitch</td>
<td>Maximum number of degrees to rotate up from straight ahead.</td>
</tr>
<tr>
<td>float TurretShapeData::minPitch</td>
<td>Minimum number of degrees to rotate down from straight ahead.</td>
</tr>
<tr>
<td>float TurretShapeData::pitchRate</td>
<td></td>
</tr>
</tbody>
</table>
Degrees per second rotation.

A value of 0 means no rotation is allowed. A value less than 0 means the rotation is instantaneous.

```cpp
bool TurretShapeData::startLoaded
```

Does the turret's mounted weapon(s) start in a loaded state.

True indicates that all mounted weapons start in a loaded state.

See also:

`ShapeBase::setImageLoaded()`

```cpp
TurretShapeFireLinkType TurretShapeData::weaponLinkType
```

Set how the mounted weapons are linked and triggered.

- **FireTogether**: All weapons fire under trigger 0.
- **GroupedFire**: Weapon mounts 0,2 fire under trigger 0, mounts 1,3 fire under trigger 1.
- **IndividualFire**: Each weapon mount fires under its own trigger 0-3.

See also:

`TurretShapeFireLinkType`

```cpp
bool TurretShapeData::zRotOnly
```

Should the turret allow only z rotations.

True indicates that the turret may only be rotated on its z axis, just like the `Item` class. This keeps the turret always upright regardless
of the surface it lands on.
Vehicle Class Reference
[ Vehicles ]

Base functionality shared by all Vehicles (FlyingVehicle, HoverVehicle, WheeledVehicle). More...

Inheritance diagram for Vehicle:

```
Legend:
SimObject
NetObject
SceneObject
GameBase
ShapeBase
Vehicle

FlyingVehicle HoverVehicle WheeledVehicle
```

List of all members.
### Public Attributes

<table>
<thead>
<tr>
<th>bool</th>
<th>disableMove</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>When this flag is set, the vehicle will ignore throttle changes.</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>static float</td>
<td>workingQueryBoxSizeMultiplier</td>
<td>How much larger the mWorkingQueryBox should be made when updating the working collision list.</td>
</tr>
<tr>
<td>static int</td>
<td>workingQueryBoxStaleThreshold</td>
<td>The maximum number of ticks that go by before the mWorkingQueryBox is considered stale and needs updating.</td>
</tr>
</tbody>
</table>
Detailed Description

Base functionality shared by all Vehicles (FlyingVehicle, HoverVehicle, WheeledVehicle).

This object implements functionality shared by all Vehicle types, but should not be instantiated directly. Create a FlyingVehicle, HoverVehicle, or WheeledVehicle instead.

Note:

The model used for any Vehicle must include a collision mesh at detail size -1.
Member Data Documentation

**bool Vehicle::disableMove**

When this flag is set, the vehicle will ignore throttle changes.

**float Vehicle::workingQueryBoxSizeMultiplier [static]**

How much larger the mWorkingQueryBox should be made when updating the working collision list.

The larger this number the less often the working list will be updated due to motion, but any non-static shape that moves into the query box will not be noticed.

**int Vehicle::workingQueryBoxStaleThreshold [static]**

The maximum number of ticks that go by before the mWorkingQueryBox is considered stale and needs updating.

Other factors can cause the collision working query box to become invalidated, such as the vehicle moving far enough outside of this cached box. The smaller this number, the more times the working list of triangles that are considered for collision is refreshed. This has the greatest impact with colliding with high triangle count meshes.

**Note:**

Set to -1 to disable any time-based forced check.
VehicleData Class Reference
[ Vehicles ]

Base properties shared by all Vehicles (FlyingVehicle, HoverVehicle, WheeledVehicle). More...

Inheritance diagram for VehicleData:

List of all members.
Public Member Functions

Callbacks

<table>
<thead>
<tr>
<th>void</th>
<th>onEnterLiquid (Vehicle obj, float coverage, string type)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Called when the vehicle enters liquid.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>onLeaveLiquid (Vehicle obj, string type)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Called when the vehicle leaves liquid.</td>
</tr>
</tbody>
</table>
### Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>bodyFriction</code></td>
<td>Collision friction coefficient.</td>
</tr>
<tr>
<td>float</td>
<td><code>bodyRestitution</code></td>
<td>Collision 'bounciness'.</td>
</tr>
<tr>
<td>float</td>
<td><code>cameraDecay</code></td>
<td>How quickly the camera moves back towards the vehicle when stopped.</td>
</tr>
<tr>
<td>float</td>
<td><code>cameraLag</code></td>
<td>How much the camera lags behind the vehicle depending on vehicle speed.</td>
</tr>
<tr>
<td>float</td>
<td><code>cameraOffset</code></td>
<td>Vertical (Z axis) height of the camera above the vehicle.</td>
</tr>
<tr>
<td>bool</td>
<td><code>cameraRoll</code></td>
<td>If true, the camera will roll with the vehicle. If false, the camera will always have the positive Z axis as up.</td>
</tr>
<tr>
<td>float</td>
<td><code>collDamageMultiplier</code></td>
<td>Damage to this vehicle after a collision (multiplied by collision velocity).</td>
</tr>
<tr>
<td>float</td>
<td><code>collDamageThresholdVel</code></td>
<td>Minimum collision velocity to cause damage to this vehicle.</td>
</tr>
<tr>
<td>float</td>
<td><code>collisionTol</code></td>
<td>Minimum distance between objects for them to be considered as colliding.</td>
</tr>
<tr>
<td>float</td>
<td><code>contactTol</code></td>
<td>Maximum relative velocity between objects for collisions to be resolved as contacts.</td>
</tr>
</tbody>
</table>

**ParticleEmitterData**  
`damageEmitter` [3]
Array of particle emitters used to generate damage (dust, smoke etc) effects.

### Point3F  damageEmitterOffset [2]
Object space "x y z" offsets used to emit particles for the active damageEmitter.

### float  damageLevelTolerance [2]
Damage levels (as a percentage of maxDamage) above which to begin emitting particles from the associated damageEmitter.

<table>
<thead>
<tr>
<th>ParticleEmitterData</th>
<th>dustEmitter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Dust particle emitter.</td>
</tr>
</tbody>
</table>

### float  dustHeight
Height above ground at which to emit particles from the dustEmitter.

<table>
<thead>
<tr>
<th>SFXProfile</th>
<th>exitingWater</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sound to play when exiting the water.</td>
</tr>
</tbody>
</table>

### float  exitSplashSoundVelocity
Minimum velocity when leaving the water for the exitingWater sound to play.

<table>
<thead>
<tr>
<th>SFXProfile</th>
<th>hardImpactSound</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sound to play on a 'hard' impact.</td>
</tr>
</tbody>
</table>

### float  hardImpactSpeed
Minimum collision speed for the hardImpactSound to be played.

### float  hardSplashSoundVelocity
Minimum velocity when entering the water for the imapactWaterHard sound to play.

<table>
<thead>
<tr>
<th>SFXProfile</th>
<th>impactWaterEasy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sound to play when entering the water with speed &gt;= softSplashSoundVelocity and &lt; mediumSplashSoundVelocity.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SFXProfile</th>
<th>impactWaterHard</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sound to play when entering the water with speed $\geq$ hardSplashSoundVelocity.</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td></td>
</tr>
<tr>
<td><strong>SFXProfile</strong> impactWaterMedium</td>
<td>Sound to play when entering the water with speed $\geq$ mediumSplashSoundVelocity and $&lt;$ hardSplashSoundVelocity.</td>
</tr>
<tr>
<td><strong>int</strong> integration</td>
<td>Number of integration steps per tick.</td>
</tr>
<tr>
<td><strong>float</strong> jetEnergyDrain</td>
<td>Energy amount to drain for each tick the vehicle is jetting.</td>
</tr>
<tr>
<td><strong>float</strong> jetForce</td>
<td>Additional force applied to the vehicle when it is jetting.</td>
</tr>
<tr>
<td><strong>Point3F</strong> massBox</td>
<td>Define the box used to estimate the vehicle's moment of inertia.</td>
</tr>
<tr>
<td><strong>Point3F</strong> massCenter</td>
<td>Defines the vehicle's center of mass (offset from the origin of the model).</td>
</tr>
<tr>
<td><strong>float</strong> maxDrag</td>
<td>Maximum drag coefficient.</td>
</tr>
<tr>
<td><strong>float</strong> maxSteeringAngle</td>
<td>Maximum yaw (horizontal) and pitch (vertical) steering angle in radians.</td>
</tr>
<tr>
<td><strong>float</strong> mediumSplashSoundVelocity</td>
<td>Minimum velocity when entering the water for the imapactWaterMedium sound to play.</td>
</tr>
<tr>
<td><strong>float</strong> minDrag</td>
<td>Minimum drag coefficient.</td>
</tr>
<tr>
<td><strong>float</strong> minImpactSpeed</td>
<td>Minimum collision speed for the onImpact</td>
</tr>
</tbody>
</table>
callback to be invoked.

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><strong>minJetEnergy</strong></td>
<td>Minimum vehicle energy level to begin jetting.</td>
</tr>
<tr>
<td>float</td>
<td><strong>minRollSpeed</strong></td>
<td>Unused.</td>
</tr>
<tr>
<td>float</td>
<td><strong>numDmgEmitterAreas</strong></td>
<td>Number of damageEmitterOffset values to use for each damageEmitter.</td>
</tr>
<tr>
<td>SFXProfile</td>
<td><strong>softImpactSound</strong></td>
<td>Sound to play on a 'soft' impact.</td>
</tr>
<tr>
<td>float</td>
<td><strong>softImpactSpeed</strong></td>
<td>Minimum collision speed for the softImpactSound to be played.</td>
</tr>
<tr>
<td>float</td>
<td><strong>softSplashSoundVelocity</strong></td>
<td>Minimum velocity when entering the water for the imapactWaterEasy sound to play.</td>
</tr>
<tr>
<td>ParticleEmitterData</td>
<td><strong>splashEmitter</strong> [2]</td>
<td>Array of particle emitters used to generate splash effects.</td>
</tr>
<tr>
<td>float</td>
<td><strong>splashFreqMod</strong></td>
<td>Number of splash particles to generate based on vehicle speed.</td>
</tr>
<tr>
<td>float</td>
<td><strong>splashVelEpsilon</strong></td>
<td>Minimum speed when moving through water to generate splash particles.</td>
</tr>
<tr>
<td>float</td>
<td><strong>triggerDustHeight</strong></td>
<td>Maximum height above surface to emit dust particles.</td>
</tr>
<tr>
<td>SFXProfile</td>
<td><strong>waterWakeSound</strong></td>
<td>Looping sound to play while moving through the water.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Base properties shared by all Vehicles (*FlyingVehicle*, *HoverVehicle*, *WheeledVehicle*).

This datablock defines properties shared by all Vehicle types, but should not be instantiated directly. Instead, set the desired properties in the *FlyingVehicleData*, *HoverVehicleData* or *WheeledVehicleData* datablock.
Damage

The VehicleData class extends the basic energy/damage functionality provided by ShapeBaseData to include damage from collisions, as well as particle emitters activated automatically when damage levels reach user specified thresholds.

The example below shows how to setup a Vehicle to:

- take damage when colliding with another object
- emit gray smoke particles from two locations on the Vehicle when damaged above 50%
- emit black smoke particles from two locations on the Vehicle when damaged above 85%
- emit bubbles when any active damage emitter point is underwater

Example:

```csharp
// damage from collisions
collDamageMultiplier = 0.05;
collDamageThresholdVel = 15;

// damage levels
damageLevelTolerance[0] = 0.5;
damageEmitter[0] = GraySmokeEmitter; //
damageLevelTolerance[1] = 0.85;
damageEmitter[1] = BlackSmokeEmitter; //
damageEmitter[2] = DamageBubbleEmitter; //

// emit offsets (used for all active damage)
damageEmitterOffset[0] = "0.5 3 1";
damageEmitterOffset[1] = "-0.5 3 1";
numDmgEmitterAreas = 2;
```
Member Function Documentation

```cpp
void VehicleData::onEnterLiquid(Vehicle obj,  
    float coverage,  
    string type)
```

Called when the vehicle enters liquid.

**Parameters:**

- `obj` the `Vehicle` object
- `coverage` percentage of the vehicle's bounding box covered by the liquid
- `type` type of liquid the vehicle has entered

```cpp
void VehicleData::onLeaveLiquid(Vehicle obj,  
    string type)
```

Called when the vehicle leaves liquid.

**Parameters:**

- `obj` the `Vehicle` object
- `type` type of liquid the vehicle has left
Member Data Documentation

**float VehicleData::bodyFriction**

Collision friction coefficient.

How well this object will slide against objects it collides with.

**float VehicleData::bodyRestitution**

Collision 'bounciness'.

Normally in the range 0 (not bouncy at all) to 1 (100% bounciness).

**float VehicleData::cameraDecay**

How quickly the camera moves back towards the vehicle when stopped.

**See also:**

cameraLag.

**float VehicleData::cameraLag**

How much the camera lags behind the vehicle depending on vehicle speed.

Increasing this value will make the camera fall further behind the vehicle as it accelerates away.

**See also:**

cameraDecay.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>VehicleData::cameraOffset</code></td>
<td>Vertical (Z axis) height of the camera above the vehicle.</td>
</tr>
<tr>
<td><code>VehicleData::cameraRoll</code></td>
<td>If true, the camera will roll with the vehicle. If false, the camera will always have the positive Z axis as up.</td>
</tr>
<tr>
<td><code>VehicleData::collDamageMultiplier</code></td>
<td>Damage to this vehicle after a collision (multiplied by collision velocity). Currently unused.</td>
</tr>
<tr>
<td><code>VehicleData::collDamageThresholdVel</code></td>
<td>Minimum collision velocity to cause damage to this vehicle. Currently unused.</td>
</tr>
<tr>
<td><code>VehicleData::collisionTol</code></td>
<td>Minimum distance between objects for them to be considered as colliding.</td>
</tr>
<tr>
<td><code>VehicleData::contactTol</code></td>
<td>Maximum relative velocity between objects for collisions to be</td>
</tr>
</tbody>
</table>
resolved as contacts.

Velocities greater than this are handled as collisions.

ParticleEmitterData VehicleData::damageEmitter[3]

Array of particle emitters used to generate damage (dust, smoke etc) effects.

Currently, the first two emitters (indices 0 and 1) are used when the damage level exceeds the associated damageLevelTolerance. The 3rd emitter is used when the emitter point is underwater.

See also:

damageEmitterOffset

Point3F VehicleData::damageEmitterOffset[2]

Object space "x y z" offsets used to emit particles for the active damageEmitter.

Example:

```cpp
// damage levels
damageLevelTolerance[0] = 0.5;
damageEmitter[0] = SmokeEmitter;
// emit offsets (used for all active damageEmitter)
damageEmitterOffset[0] = "0.5 3 1";
damageEmitterOffset[1] = "-0.5 3 1";
numDmgEmitterAreas = 2;
```

float VehicleData::damageLevelTolerance[2]

Damage levels (as a percentage of maxDamage) above which to
begin emitting particles from the associated damageEmitter.

Levels should be in order of increasing damage.

**See also:**
- damageEmitterOffset

**ParticleEmitterData VehicleData::dustEmitter**

Dust particle emitter.

**See also:**
- triggerDustHeight
- dustHeight

**float VehicleData::dustHeight**

Height above ground at which to emit particles from the dustEmitter.

**SFXProfile VehicleData::exitingWater**

Sound to play when exiting the water.

**float VehicleData::exitSplashSoundVelocity**

Minimum velocity when leaving the water for the exitingWater sound to play.

**SFXProfile VehicleData::hardImpactSound**
Sound to play on a 'hard' impact.

This sound is played if the impact speed \( \geq \) hardImpactSpeed.

**See also:**

- hardImpactSpeed

---

**float VehicleData::hardImpactSpeed**

Minimum collision speed for the hardImpactSound to be played.

---

**float VehicleData::hardSplashSoundVelocity**

Minimum velocity when entering the water for the imapactWaterHard sound to play.

**See also:**

- impactWaterHard

---

**SFXProfile VehicleData::impactWaterEasy**

Sound to play when entering the water with speed \( \geq \) softSplashSoundVelocity and \( < \) mediumSplashSoundVelocity.

---

**SFXProfile VehicleData::impactWaterHard**

Sound to play when entering the water with speed \( \geq \) hardSplashSoundVelocity.

---

**SFXProfile VehicleData::impactWaterMedium**
Sound to play when entering the water with speed \( \geq \) mediumSplashSoundVelocity and \( < \) hardSplashSoundVelocity.

**int VehicleData::integration**

Number of integration steps per tick.

Increase this to improve simulation stability (also increases simulation processing time).

**float VehicleData::jetEnergyDrain**

Energy amount to drain for each tick the vehicle is jetting.

Once the vehicle's energy level reaches 0, it will no longer be able to jet.

**float VehicleData::jetForce**

Additional force applied to the vehicle when it is jetting.

For WheeledVehicles, the force is applied in the forward direction. For FlyingVehicles, the force is applied in the thrust direction.

**Point3F VehicleData::massBox**

Define the box used to estimate the vehicle's moment of inertia.

Currently only used by WheeledVehicle; other vehicle types use a unit sphere to compute inertia.

**Point3F VehicleData::massCenter**
Defines the vehicle's center of mass (offset from the origin of the model).

```
float VehicleData::maxDrag
```

Maximum drag coefficient.
Currently unused.

```
float VehicleData::maxSteeringAngle
```

Maximum yaw (horizontal) and pitch (vertical) steering angle in radians.

```
float VehicleData::mediumSplashSoundVelocity
```

Minimum velocity when entering the water for the impactWaterMedium sound to play.

**See also:**

impactWaterMedium

```
float VehicleData::minDrag
```

Minimum drag coefficient.
Currently only used by FlyingVehicle.

```
float VehicleData::minImpactSpeed
```

Minimum collision speed for the onImpact callback to be invoked.
**float VehicleData::minJetEnergy**

Minimum vehicle energy level to begin jetting.

**float VehicleData::minRollSpeed**

Unused.

**float VehicleData::numDmgEmitterAreas**

Number of damageEmitterOffset values to use for each damageEmitter.

*See also:*  
`damageEmitterOffset`

**SFXProfile VehicleData::softImpactSound**

Sound to play on a 'soft' impact.

This sound is played if the impact speed is `< hardImpactSpeed` and `>= softImpactSpeed`.

*See also:*  
`softImpactSpeed`

**float VehicleData::softImpactSpeed**

Minimum collision speed for the softImpactSound to be played.
**float VehicleData::softSplashSoundVelocity**

Minimum velocity when entering the water for the `impactWaterEasy` sound to play.

**See also:**

`impactWaterEasy`

---

**ParticleEmitterData VehicleData::splashEmitter[2]**

Array of particle emitters used to generate splash effects.

---

**float VehicleData::splashFreqMod**

Number of splash particles to generate based on vehicle speed.

This value is multiplied by the current speed to determine how many particles to generate each frame.

---

**float VehicleData::splashVelEpsilon**

Minimum speed when moving through water to generate splash particles.

---

**float VehicleData::triggerDustHeight**

Maximum height above surface to emit dust particles.

If the vehicle is less than `triggerDustHeight` above a static surface with a material that has 'showDust' set to true, the vehicle will emit particles from the `dustEmitter`. 
SFXProfile VehicleData::waterWakeSound

Looping sound to play while moving through the water.
**WaterBlock Class Reference**

**[Water]**

A block shaped water volume defined by a 3D scale and orientation.

*More...*

Inheritance diagram for WaterBlock:

```
+---------------------------+       +---------------------------+
| SimObject                |       | NetObject                 |
|                          |       |                           |
|                          +-------+---------------------------+
|                           |       | SceneObject               |
|                          +-------+---------------------------+
|                           |       | WaterObject               |
|                          +-------+---------------------------+
|                           |       | WaterBlock               |
```

[legend]

*List of all members.*
## Public Attributes

**WaterBlock**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>gridElementSize</td>
<td>Spacing between vertices in the WaterBlock mesh.</td>
</tr>
<tr>
<td>float</td>
<td>gridSize</td>
<td>Duplicate of gridElementSize for backwards compatibility.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>static bool</th>
<th>isRenderable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static bool</th>
<th>isSelectable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

A block shaped water volume defined by a 3D scale and orientation.

See also:

WaterObject for inherited functionality.
Member Data Documentation

**float WaterBlock::gridElementSize**

Spacing between vertices in the *WaterBlock* mesh.

**float WaterBlock::gridSize**

Duplicate of gridElementSize for backwards compatibility.
**WaterObject Class Reference**

[Water]

Abstract base class for representing a body of water. More...

Inheritance diagram for WaterObject:

```
  +------------------+
  | SimObject        |
  +------------------+
        ^          ^
        |          |
        +----------+
        |          |
        +----------+
  +------------------+
  | NetObject        |
  +------------------+
        ^          ^
        |          |
        +----------+
        |          |
        +----------+
  +------------------+
  | SceneObject      |
  +------------------+
        ^          ^
        |          |
        +----------+
        |          |
        +----------+
  +------------------+
  | WaterObject      |
  +------------------+
```

River  WaterBlock  WaterPlane

[legend]

List of all members.
### Public Attributes

#### WaterObject

<table>
<thead>
<tr>
<th>Colorl</th>
<th>baseColor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Changes color of water fog.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>density</th>
</tr>
</thead>
<tbody>
<tr>
<td>Affects buoyancy of an object, thus affecting the Z velocity of a player (jumping, falling, etc.).</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>emissive</th>
</tr>
</thead>
<tbody>
<tr>
<td>When true the water colors don't react to changes to environment lighting.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>foamAmbientLerp</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Point2F</th>
<th>foamDir [2]</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>foamMaxDepth</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>foamOpacity [2]</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>foamRippleInfluence</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>foamSpeed [2]</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>filename</th>
<th>foamTex</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diffuse texture for foam in shallow water (advanced lighting only).</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Point2F</th>
<th>foamTexScale [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td>applied to the surface.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>fresnelBias</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extent of fresnel affecting reflection fogging.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>fresnelPower</th>
</tr>
</thead>
<tbody>
<tr>
<td>Measures intensity of affect on reflection based on fogging.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string</th>
<th>liquidType</th>
</tr>
</thead>
<tbody>
<tr>
<td>Liquid type of <strong>WaterBlock</strong>, such as water, ocean,</td>
<td></td>
</tr>
</tbody>
</table>
Currently only Water is defined and used.

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>overallFoamOpacity</code></td>
<td>Master variable affecting entire surface.</td>
</tr>
<tr>
<td>float</td>
<td><code>overallRippleMagnitude</code></td>
<td>Master variable affecting entire body of water's undulation.</td>
</tr>
<tr>
<td>float</td>
<td><code>overallWaveMagnitude</code></td>
<td>Master variable affecting entire body of water's undulation.</td>
</tr>
<tr>
<td>float</td>
<td><code>rippleMagnitude</code> [3]</td>
<td>Intensifies the vertex modification of the surface.</td>
</tr>
<tr>
<td>filename</td>
<td><code>rippleTex</code></td>
<td>Normal map used to simulate small surface ripples.</td>
</tr>
<tr>
<td>Point2F</td>
<td><code>rippleTexScale</code> [3]</td>
<td>Intensifies the affect of the normal map applied to the surface.</td>
</tr>
<tr>
<td>ColorF</td>
<td><code>specularColor</code></td>
<td>Color used for specularity on the water surface (sun only).</td>
</tr>
<tr>
<td>float</td>
<td><code>specularPower</code></td>
<td>Power used for specularity on the water surface (sun only).</td>
</tr>
<tr>
<td>float</td>
<td><code>viscosity</code></td>
<td>Affects drag force applied to an object submerged in this container.</td>
</tr>
<tr>
<td>float</td>
<td><code>waveMagnitude</code> [3]</td>
<td></td>
</tr>
</tbody>
</table>
Height of water undulation.


**Basic Lighting**

<table>
<thead>
<tr>
<th>float</th>
<th>clarity</th>
<th>Relative opacity or transparency of the water surface.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color</td>
<td>underwaterColor</td>
<td>Changes the color shading of objects beneath the water surface.</td>
</tr>
</tbody>
</table>

**Reflect**

| string  | cubemap | Cubemap used instead of reflection texture if fullReflect is off. |
| bool    | fullReflect | Enables dynamic reflection rendering. |
| float   | reflectDetailAdjust | scale up or down the detail level for objects rendered in a reflection |
| float   | reflectivity | Overall scalar to the reflectivity of the water surface. |
| int     | reflectMaxRateMs | Affects the sort time of reflected objects. |
| bool    | reflectNormalUp | always use z up as the reflection normal |
| float   | reflectPriority | Affects the sort order of reflected objects. |
**int reflectTexSize**
The texture size used for reflections (square).

**bool useOcclusionQuery**
turn off reflection rendering when occluded (delayed).

**Misc**

**float depthGradientMax**
Depth in world units, the max range of the color gradient texture.

**filename depthGradientTex**
1D texture defining the base water color by depth

**Distortion**

**float distortEndDist**
Max distance that distortion algorithm is performed. The lower, the more distorted the effect.

**float distortFullDepth**
Determines the scaling down of distortion in shallow water.

**float distortStartDist**
Determines start of distortion effect where water surface intersects the camera near plane.

**Sound**

**SFXAmbience soundAmbience**
Ambient sound environment when listener is submerged.

**Underwater Fogging**
<table>
<thead>
<tr>
<th>float</th>
<th><code>waterFogDensity</code></th>
<th>Intensity of underwater fogging.</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>waterFogDensityOffset</code></td>
<td>Delta, or limit, applied to <code>waterFogDensity</code>.</td>
</tr>
<tr>
<td>float</td>
<td><code>wetDarkening</code></td>
<td>The refract color intensity scaled at <code>wetDepth</code>.</td>
</tr>
<tr>
<td>float</td>
<td><code>wetDepth</code></td>
<td>The depth in world units at which full darkening will be received, giving a wet look to objects underwater.</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>static bool</th>
<th>isRenderable</th>
<th>Disables rendering of all instances of this type.</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>wireframe</td>
<td>If true, will render the wireframe of the WaterObject.</td>
</tr>
</tbody>
</table>
Detailed Description

Abstract base class for representing a body of water.

WaterObject is abstract and may not be created. It defines functionality shared by its derived classes.

WaterObject exposes many fields for controlling it visual quality.

WaterObject surface rendering has the following general features:

- Waves represented by vertex undulation and user parameters.
- Ripples represented by a normal map and user parameters.
- Refraction of underwater objects.
- Dynamic planar reflection or static cubemap reflection.
- Paramable water fog and color shift.

It will, however, look significantly different depending on the LightingManager that is active. With Basic Lighting, we do not have a prepass texture to lookup per-pixel depth and therefore cannot use our rendering techniques that depend on it.

In particular, the following field groups are not used under Basic Lighting:

- Underwater Fogging
- Misc
- Distortion
- And foam related fields under the WaterObject group.

WaterObject also defines several fields for gameplay use and objects that support buoyancy.
## Member Data Documentation

<table>
<thead>
<tr>
<th>Member Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorI</td>
<td>WaterObject::baseColor</td>
<td>Changes color of water fog.</td>
</tr>
<tr>
<td>float</td>
<td>WaterObject::clarity</td>
<td>Relative opacity or transparency of the water surface.</td>
</tr>
<tr>
<td>string</td>
<td>WaterObject::cubemap</td>
<td>Cubemap used instead of reflection texture if fullReflect is off.</td>
</tr>
<tr>
<td>float</td>
<td>WaterObject::density</td>
<td>Affects buoyancy of an object, thus affecting the Z velocity of a player (jumping, falling, etc.)</td>
</tr>
<tr>
<td>float</td>
<td>WaterObject::depthGradientMax</td>
<td>Depth in world units, the max range of the color gradient texture.</td>
</tr>
<tr>
<td>filename</td>
<td>WaterObject::depthGradientTex</td>
<td>1D texture defining the base water color by depth</td>
</tr>
<tr>
<td>float</td>
<td>WaterObject::distortEndDist</td>
<td></td>
</tr>
</tbody>
</table>
Max distance that distortion algorithm is performed. The lower, the more distorted the effect.

float WaterObject::distortFullDepth

Determines the scaling down of distortion in shallow water.

float WaterObject::distortStartDist

Determines start of distortion effect where water surface intersects the camera near plane.

bool WaterObject::emissive

When true the water colors don't react to changes to environment lighting.

float WaterObject::foamAmbientLerp

Point2F WaterObject::foamDir[2]

float WaterObject::foamMaxDepth

float WaterObject::foamOpacity[2]

float WaterObject::foamRippleInfluence

float WaterObject::foamSpeed[2]
<table>
<thead>
<tr>
<th><strong>filename</strong> WaterObject::foamTex</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diffuse texture for foam in shallow water (advanced lighting only).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Point2F WaterObject::foamTexScale[2]</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>applied to the surface.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float WaterObject::fresnelBias</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Extent of fresnel affecting reflection fogging.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float WaterObject::fresnelPower</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Measures intensity of affect on reflection based on fogging.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>bool WaterObject::fullReflect</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Enables dynamic reflection rendering.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>string WaterObject::liquidType</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Liquid type of WaterBlock, such as water, ocean, lava. Currently only Water is defined and used.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>float WaterObject::overallFoamOpacity</strong></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th><strong>float WaterObject::overallRippleMagnitude</strong></th>
</tr>
</thead>
</table>
Master variable affecting entire surface.

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>WaterObject::overallWaveMagnitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>Master variable affecting entire body of water's undulation.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>WaterObject::reflectDetailAdjust</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scale up or down the detail level for objects rendered in a reflection</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>WaterObject::reflectivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall scalar to the reflectivity of the water surface.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>WaterObject::reflectMaxRateMs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Affects the sort time of reflected objects.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>WaterObject::reflectNormalUp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Always use z up as the reflection normal</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>WaterObject::reflectPriority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Affects the sort order of reflected objects.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>WaterObject::reflectTexSize</th>
</tr>
</thead>
</table>
The texture size used for reflections (square).

**Point2F WaterObject::rippleDir[3]**

Modifies the direction of ripples on the surface.

**float WaterObject::rippleMagnitude[3]**

Intensifies the vertext modification of the surface.

**float WaterObject::rippleSpeed[3]**

Modifies speed of surface ripples.

**filename WaterObject::rippleTex**

Normal map used to simulate small surface ripples.

**Point2F WaterObject::rippleTexScale[3]**

Intensifies the affect of the normal map applied to the surface.

**SFXAmbience WaterObject::soundAmbience**

Ambient sound environment when listener is submerged.

**ColorF WaterObject::specularColor**
Color used for specularity on the water surface (sun only).

**float WaterObject::specularPower**

Power used for specularity on the water surface (sun only).

**ColorI WaterObject::underwaterColor**

Changes the color shading of objects beneath the water surface.

**bool WaterObject::useOcclusionQuery**

Turn off reflection rendering when occluded (delayed).

**float WaterObject::viscosity**

Affects drag force applied to an object submerged in this container.

**float WaterObject::waterFogDensity**

Intensity of underwater fogging.

**float WaterObject::waterFogDensityOffset**

Delta, or limit, applied to waterFogDensity.

**Point2F WaterObject::waveDir[3]**
Direction waves flow toward shores.

float WaterObject::waveMagnitude[3]

Height of water undulation.

float WaterObject::waveSpeed[3]

Speed of water undulation.

float WaterObject::wetDarkening

The refract color intensity scaled at wetDepth.

float WaterObject::wetDepth

The depth in world units at which full darkening will be received, giving a wet look to objects underwater.
WaterPlane Class Reference

[Water]

Represents a large body of water stretching to the horizon in all directions. More...

Inheritance diagram for WaterPlane:

```
SimObject

NetObject

SceneObject

WaterObject

WaterPlane
```

List of all members.
Public Attributes

**WaterPlane**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>gridElementSize</td>
<td>Duplicate of gridElementSize for backwards compatibility.</td>
</tr>
<tr>
<td>int</td>
<td>gridSize</td>
<td>Spacing between vertices in the WaterBlock mesh.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

Represents a large body of water stretching to the horizon in all directions.

WaterPlane's position is defined only height, the z element of position, it is infinite in xy and depth. WaterPlane is designed to represent the ocean on an island scene and viewed from ground level; other uses may not be appropriate and a WaterBlock may be used.

See also:

   WaterObject for inherited functionality.

Limitations:

Because WaterPlane cannot be projected exactly to the far-clip distance, other objects nearing this distance can have noticeable artifacts as they clip through first the WaterPlane and then the far plane.

To avoid this large objects should be positioned such that they will not line up with the far-clip from vantage points the player is expected to be. In particular, your TerrainBlock should be completely contained by the far-clip distance.

Viewing WaterPlane from a high altitude with a tight far-clip distance will accentuate this limitation. WaterPlane is primarily designed to be viewed from ground level.
Member Data Documentation

float WaterPlane::gridElementSize

Duplicate of gridElementSize for backwards compatibility.

int WaterPlane::gridSize

Spacing between vertices in the WaterBlock mesh.
WayPoint Class Reference

[Miscellaneous]

Special type of marker, distinguished by a name and team ID number. More...

Inheritance diagram for WayPoint:

```
+-- SimObject
    +-- NetObject
    |    +-- SceneObject
    |        +-- GameBase
    |            +-- ShapeBase
    |                +-- MissionMarker
    |                    +-- WayPoint
```

List of all members.
## Public Attributes

### Misc

<table>
<thead>
<tr>
<th>caseString</th>
<th>markerName</th>
<th>Unique name representing this waypoint.</th>
</tr>
</thead>
<tbody>
<tr>
<td>WayPointTeam</td>
<td>team</td>
<td>Unique numerical ID assigned to this waypoint, or set of waypoints.</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
**Detailed Description**

Special type of marker, distinguished by a name and team ID number.

The original Torque engines were built from a multi-player game called Tribes. The Tribes series featured various team based game modes, such as capture the flag. The `WayPoint` class survived the conversion from game (Tribes) to game engine (Torque).

Essentially, this is a `MissionMarker` with the addition of two variables: `markerName` and `team`. Whenever a `WayPoint` is created, it is added to a unique global list called `WayPointSet`. You can iterate through this set, seeking out specific markers determined by their `markerName` and `team` ID. This avoids the overhead of constantly calling `commandToClient` and `commandToServer` to determine a `WayPoint` object's name, unique ID, etc.

**Note:**

The `markerName` field was previously called `name`, but was changed because this conflicted with the `SimObject` name field. Existing scripts that relied on the `WayPoint` name field will need to be updated.

**Example:**

```plaintext
new WayPoint()
{
    team = "1";
    dataBlock = "WayPointMarker";
    position = "-0.0224786 1.53471 2.93219"
    rotation = "1 0 0 0"
    scale = "1 1 1"
    canSave = "1"
    canSaveDynamicFields = "1"
};
```
See also:

MissionMarker
MissionMarkerData
### Member Data Documentation

<table>
<thead>
<tr>
<th>Member Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>caseString WayPoint::markerName</code></td>
<td>Unique name representing this waypoint.</td>
</tr>
<tr>
<td><code>WayPointTeam WayPoint::team</code></td>
<td>Unique numerical ID assigned to this waypoint, or set of waypoints.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
WheeledVehicle Class Reference
[Vehicles]

A wheeled vehicle. More...

Inheritance diagram for WheeledVehicle:

List of all members.
Public Member Functions

<table>
<thead>
<tr>
<th>int</th>
<th>getWheelCount ()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Get the number of wheels on this vehicle.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>setWheelPowered (int wheel, bool powered)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Set whether the wheel is powered (has torque applied from the engine).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>setWheelSpring (int wheel, WheeledVehicleSpring spring)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Set the WheeledVehicleSpring datablock for this wheel.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>setWheelSteering (int wheel, float steering)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Set how much the wheel is affected by steering.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bool</th>
<th>setWheelTire (int wheel, WheeledVehicleTire tire)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Set the WheeledVehicleTire datablock for this wheel.</td>
</tr>
</tbody>
</table>
### Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td>isRenderable</td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td>isSelectable</td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
**Detailed Description**

A wheeled vehicle.

A multi-wheeled vehicle.

The model used for the **WheeledVehicle** should contain the elements shown below. Only the collision mesh and hub nodes are actually required for the object to be added to the simulation, but the suspension will look strange if the spring animations are not present.

**Collision mesh**

A convex collision mesh at detail size -1.

**Hub nodes**

The model must contain a node for each wheel called hubN, where N is an integer value starting from 0. For example, a four wheeled vehicle would have nodes: hub0, hub1, hub2, and hub3. The wheel model (specified by **WheeledVehicleTire**) is positioned at the hub node, and automatically rotated to the right orientation (whether on the left or right side of the vehicle).

**Spring animations**

To visualise the suspension action, the vehicle model should contain a non-cyclic animation sequence for each wheel that animates the appropriate hub node from t=0 (fully compressed to t=1 (fully extended). The sequences must be called springN, where N matches the wheel hub index.

**Steering animation**

Optional non-cyclic animation called 'steering' that animates from t=0 (full right) to t=0.5 (center) to t=1 (full left).

**Brakelight animation**

Optional non-cyclic animation called 'brakeLight' that animates from t=0 (off) to t=1 (braking). This is usually a 2-frame animation controlling the visibility of a quad or mesh to represent each brake light.

"
The example below shows the datablocks required for a simple 4-wheeled vehicle. The script should be executed on the server, and the vehicle can then be added to the simulation programmatically from the level startup scripts, or by selecting the MyCar datablock from the World Editor (Library->ScriptedObjects->Vehicles).

Example:

```plaintext
datablock WheeledVehicleTire( MyCarTire )
{
    shapeFile = "art/shapes/wheel.dts";
    staticFriction = 4.2;
    kineticFriction = 1.0;

    lateralForce = 18000;
    lateralDamping = 6000;
    lateralRelaxation = 1;

    longitudinalForce = 18000;
    longitudinalDamping = 4000;
    longitudinalRelaxation = 1;
    radius = 0.61;
};

datablock WheeledVehicleSpring( MyCarSpring )
{
    length = 0.5;
    force = 2800;
    damping = 3600;
    antiSwayForce = 3;
};

datablock WheeledVehicleData( MyCar )
{
    category = "Vehicles";
```
shapeFile = "art/shapes/car.dts";

maxSteeringAngle = 0.585;

// 3rd person camera settings
cameraRoll = false;
cameraMaxDist = 7.8;
cameraOffset = 1.0;
cameraLag = 0.3;
cameraDecay = 1.25;

useEyePoint = true;

// Rigid Body
mass = "400";
massCenter = "0 -0.2 0";
massBox = "0 0 0";

drag = 0.6;
bodyFriction = 0.6;
bodyRestitution = 0.4;
minImpactSpeed = 5;
softImpactSpeed = 5;
hardImpactSpeed = 15;
integration = 8;
collisionTol = 0.05;
contactTol = 0.4;

// Engine
engineTorque = 4300;
engineBrake = 5000;
brakeTorque = 10000;
maxWheelSpeed = 50;
// Energy
maxEnergy = 100;
jetForce = 3000;
minJetEnergy = 30;
jetEnergyDrain = 2;

// Sounds
engineSound = CarEngineSnd;
squealSound = CarSquealSnd;
softImpactSound = SoftImpactSnd;
hardImpactSound = HardImpactSnd;

// Particles
tireEmitter = "CarTireEmitter";
dustEmitter = "CarTireEmitter";
dustHeight = "1";
}

// This function is executed when the WheeledVehicle object is added to the simulation.
function MyCar::onAdd( %this, %obj )
{
    Parent::onAdd( %this, %obj );

    // Setup the car with some tires & springs
    for ( %i = %obj.getWheelCount() - 1; %i >= 0; %i-- )
    {
        %obj.setWheelTire( %i, MyCarTire );
        %obj.setWheelSpring( %i, MyCarSpring );
        %obj.setWheelPowered( %i, true );
    }
// Steer with the front tires only
%obj.setWheelSteering( 0, 1 );
%obj.setWheelSteering( 1, 1 );
}
**Member Function Documentation**

```cpp
int WheeledVehicle::getWheelCount()
```

Get the number of wheels on this vehicle.

**Returns:**
the number of wheels (equal to the number of hub nodes defined in the model)

```cpp
bool WheeledVehicle::setWheelPowered(int wheel, bool powered)
```

Set whether the wheel is powered (has torque applied from the engine).

A rear wheel drive car for example would set the front wheels to false, and the rear wheels to true.

**Parameters:**
- `wheel` index of the wheel to set (hub node #)
- `powered` flag indicating whether to power the wheel or not

**Returns:**
true if successful, false if failed

```cpp
bool WheeledVehicle::setWheelSpring(int wheel, WheeledVehicleSpring spring)
```

Set the `WheeledVehicleSpring` datablock for this wheel.

**Parameters:**
Returns:
true if successful, false if failed

Example:
```cpp
%obj.setWheelSpring( 0, FrontSpring );
```

```
bool WheeledVehicle::setWheelSteering(int wheel, float steering )
```

Set how much the wheel is affected by steering.

The steering factor controls how much the wheel is rotated by the vehicle steering. For example, most cars would have their front wheels set to 1.0, and their rear wheels set to 0 since only the front wheels should turn.

Negative values will turn the wheel in the opposite direction to the steering angle.

Parameters:

- `wheel`  index of the wheel to set (hub node #)
- `steering` steering factor from -1 (full inverse) to 1 (full)

Returns:
true if successful, false if failed

```
bool WheeledVehicle::setWheelTire(int wheel, WheeledVehicleTire tire )
```
Set the `WheeledVehicleTire` datablock for this wheel.

**Parameters:**

- `wheel` index of the wheel to set (hub node #)
- `tire` `WheeledVehicleTire` datablock

**Returns:**

- true if successful, false if failed

**Example:**

```plaintext
%obj.setWheelTire( 0, FrontTire );
```
**WheeledVehicleData Class Reference**

[Vehicles]

Defines the properties of a **WheeledVehicle**. More...

Inheritance diagram for WheeledVehicleData:

```
  SimObject
     |       
  SimDataBlock
     |       
  GameBaseData
     |       
  ShapeBaseData
     |       
  VehicleData
     |       
WheeledVehicleData
```

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>data type</th>
<th>attribute</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>brakeTorque</code></td>
<td>Torque applied when braking.</td>
</tr>
<tr>
<td>float</td>
<td><code>engineBrake</code></td>
<td>Braking torque applied by the engine when the throttle and brake are both 0.</td>
</tr>
<tr>
<td>SFXTrack</td>
<td><code>engineSound</code></td>
<td>Looping engine sound.</td>
</tr>
<tr>
<td>float</td>
<td><code>engineTorque</code></td>
<td>Torque available from the engine at 100% throttle.</td>
</tr>
<tr>
<td>SFXTrack</td>
<td><code>jetSound</code></td>
<td>Looping sound played when the vehicle is jetting.</td>
</tr>
<tr>
<td>float</td>
<td><code>maxWheelSpeed</code></td>
<td>Maximum linear velocity of each wheel.</td>
</tr>
<tr>
<td>SFXTrack</td>
<td><code>squealSound</code></td>
<td>Looping sound played while any of the wheels is slipping.</td>
</tr>
<tr>
<td>ParticleEmitterData</td>
<td><code>tireEmitter</code></td>
<td>ParticleEmitterData datablock used to generate particles from each wheel when the vehicle is moving and the wheel is in contact with the ground.</td>
</tr>
<tr>
<td>SFXTrack</td>
<td><code>WheelImpactSound</code></td>
<td>Sound played when the wheels impact the ground.</td>
</tr>
</tbody>
</table>
Detailed Description

Defines the properties of a **WheeledVehicle**.
Member Data Documentation

**float** `WheeledVehicleData::brakeTorque`

Torque applied when braking.

This controls how fast the vehicle will stop when the brakes are applied.

**float** `WheeledVehicleData::engineBrake`

Braking torque applied by the engine when the throttle and brake are both 0.

This controls how quickly the vehicle will coast to a stop.

**SFXTrack** `WheeledVehicleData::engineSound`

Looping engine sound.

The pitch is dynamically adjusted based on the current engine RPM

**float** `WheeledVehicleData::engineTorque`

Torque available from the engine at 100% throttle.

This controls vehicle acceleration. ie. how fast it will reach maximum speed.

**SFXTrack** `WheeledVehicleData::jetSound`
Looping sound played when the vehicle is jetting.

`float WheeledVehicleData::maxWheelSpeed`

Maximum linear velocity of each wheel.
This caps the maximum speed of the vehicle.

`SFXTrack WheeledVehicleData::squealSound`

Looping sound played while any of the wheels is slipping.
The volume is dynamically adjusted based on how much the wheels are slipping.

`ParticleEmitterData WheeledVehicleData::tireEmitter`

`ParticleEmitterData` datablock used to generate particles from each wheel when the vehicle is moving and the wheel is in contact with the ground.

`SFXTrack WheeledVehicleData::WheelImpactSound`

Sound played when the wheels impact the ground.
Currently unused.
WheeledVehicleSpring Class Reference
[Vehicles]

Defines the properties of a WheeledVehicle spring. More...

Inheritance diagram for WheeledVehicleSpring:

```
SimObject

SimDataBlock

WheeledVehicleSpring
```

[legend]

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>antiSwayForce</td>
<td>Force applied to equalize extension of the spring on the opposite wheel.</td>
</tr>
<tr>
<td>float</td>
<td>damping</td>
<td>Force applied to slow changes to the extension of this spring.</td>
</tr>
<tr>
<td>float</td>
<td>force</td>
<td>Maximum spring force (when compressed to minimum length, 0).</td>
</tr>
<tr>
<td>float</td>
<td>length</td>
<td>Maximum spring length. ie. how far the wheel can extend from the root hub position.</td>
</tr>
</tbody>
</table>
Detailed Description

Defines the properties of a WheeledVehicle spring.
Member Data Documentation

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float WheeledVehicleSpring::antiSwayForce</td>
<td>Force applied to equalize extension of the spring on the opposite wheel. This force helps to keep the suspension balanced when opposite wheels are at different heights.</td>
</tr>
<tr>
<td>float WheeledVehicleSpring::damping</td>
<td>Force applied to slow changes to the extension of this spring. Increasing this makes the suspension stiffer which can help stabilise bouncy vehicles.</td>
</tr>
<tr>
<td>float WheeledVehicleSpring::force</td>
<td>Maximum spring force (when compressed to minimum length, 0). Increasing this will make the vehicle suspension ride higher (for a given vehicle mass), and also make the vehicle more bouncy when landing jumps.</td>
</tr>
<tr>
<td>float WheeledVehicleSpring::length</td>
<td>Maximum spring length. ie. how far the wheel can extend from the root hub position. This should be set to the vertical (Z) distance the hub travels in the associated spring animation.</td>
</tr>
</tbody>
</table>
WheeledVehicleTire Class Reference
[Vehicles]

Defines the properties of a WheeledVehicle tire. More...

Inheritance diagram for WheeledVehicleTire:

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>kineticFriction</code></td>
<td>Tire friction when the wheel is slipping (no traction).</td>
</tr>
<tr>
<td>float</td>
<td><code>lateralDamping</code></td>
<td>Damping force applied against lateral forces generated by the tire.</td>
</tr>
<tr>
<td>float</td>
<td><code>lateralForce</code></td>
<td>Tire force perpendicular to the direction of movement.</td>
</tr>
<tr>
<td>float</td>
<td><code>lateralRelaxation</code></td>
<td>Relaxing force applied against lateral forces generated by the tire.</td>
</tr>
<tr>
<td>float</td>
<td><code>longitudinalDamping</code></td>
<td>Damping force applied against longitudinal forces generated by the tire.</td>
</tr>
<tr>
<td>float</td>
<td><code>longitudinalForce</code></td>
<td>Tire force in the direction of movement.</td>
</tr>
<tr>
<td>float</td>
<td><code>longitudinalRelaxation</code></td>
<td>Relaxing force applied against longitudinal forces generated by the tire.</td>
</tr>
<tr>
<td>float</td>
<td><code>mass</code></td>
<td>The mass of the wheel.</td>
</tr>
<tr>
<td>float</td>
<td><code>radius</code></td>
<td>The radius of the wheel.</td>
</tr>
<tr>
<td>float</td>
<td><code>restitution</code></td>
<td>Tire restitution.</td>
</tr>
<tr>
<td>filename</td>
<td><code>shapeFile</code></td>
<td>The path to the shape to use for the wheel.</td>
</tr>
<tr>
<td>float</td>
<td><code>staticFriction</code></td>
<td>Tire friction when the wheel is not slipping (has traction).</td>
</tr>
</tbody>
</table>
Detailed Description

Defines the properties of a WheeledVehicle tire.

Tires act as springs and generate lateral and longitudinal forces to move the vehicle. These distortion/spring forces are what convert wheel angular velocity into forces that act on the rigid body.
## Member Data Documentation

### float WheeledVehicleTire::kineticFriction

Tire friction when the wheel is slipping (no traction).

### float WheeledVehicleTire::lateralDamping

Damping force applied against lateral forces generated by the tire.

**See also:**

lateralForce

### float WheeledVehicleTire::lateralForce

Tire force perpendicular to the direction of movement.

Lateral force can in simple terms be considered left/right steering force. WheeledVehicles are acted upon by forces generated by their tires and the lateralForce measures the magnitude of the force exerted on the vehicle when the tires are deformed along the x-axis. With real wheeled vehicles, tires are constantly being deformed and it is the interplay of deformation forces which determines how a vehicle moves. In Torque's simulation of vehicle physics, tire deformation obviously can't be handled with absolute realism, but the interplay of a vehicle's velocity, its engine's torque and braking forces, and its wheels' friction, lateral deformation, lateralDamping, lateralRelaxation, longitudinal deformation, longitudinalDamping, and longitudinalRelaxation forces, along with its wheels' angular velocity are combined to create a robust real-time physical simulation.

For this field, the larger the value supplied for the lateralForce, the larger the effect steering maneuvers can have. In Torque tire forces are applied at a vehicle's wheel hubs.
float WheeledVehicleTire::lateralRelaxation

Relaxing force applied against lateral forces generated by the tire.

The lateralRelaxation force measures how strongly the tire effectively un-deforms.

See also:
  lateralForce

float WheeledVehicleTire::longitudinalDamping

Damping force applied against longitudinal forces generated by the tire.

See also:
  longitudinalForce

float WheeledVehicleTire::longitudinalForce

Tire force in the direction of movement.

Longitudinal force can in simple terms be considered forward/backward movement force. WheeledVehicles are acted upon by forces generated by their tires and the longitudinalForce measures the magnitude of the force exerted on the vehicle when the tires are deformed along the y-axis.

For this field, the larger the value, the larger the effect acceleration/deceleration inputs have.

See also:
  lateralForce
float *WheeledVehicleTire::longitudinalRelaxation

Relaxing force applied against longitudinal forces generated by the tire.

The longitudinalRelaxation force measures how strongly the tire effectively un-deforms.

See also:

longitudinalForce

float *WheeledVehicleTire::mass

The mass of the wheel.

Currently unused.

float *WheeledVehicleTire::radius

The radius of the wheel.

The radius is determined from the bounding box of the shape provided in the shapefile field, and does not need to be specified in script. The tire should be built with its hub axis along the object's Y-axis.

float *WheeledVehicleTire::restitution

Tire restitution.

Currently unused.
filename `WheeledVehicleTire::shapeFile`

The path to the shape to use for the wheel.

`float WheeledVehicleTire::staticFriction`

Tire friction when the wheel is not slipping (has traction).
ZipObject Class Reference
[File I/O]

Provides access to a zip file. More...

Inheritance diagram for ZipObject:

```
  SimObject
     ^
    / 
ZipObject
```

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>addFile</code></td>
<td>(string <code>filename</code>, string <code>pathInZip</code>, bool <code>replace=true</code>) Add a file to the zip archive.</td>
</tr>
<tr>
<td>void</td>
<td><code>closeArchive</code></td>
<td>() Close an already opened zip archive.</td>
</tr>
<tr>
<td>void</td>
<td><code>closeFile</code></td>
<td>(SimObject <code>stream</code>) Close a previously opened file within the zip archive.</td>
</tr>
<tr>
<td>bool</td>
<td><code>deleteFile</code></td>
<td>(string <code>pathInZip</code>) Deleted the given file from the zip archive.</td>
</tr>
<tr>
<td>bool</td>
<td><code>extractFile</code></td>
<td>(string <code>pathInZip</code>, string <code>filename</code>) Extact a file from the zip archive and save it to the requested location.</td>
</tr>
<tr>
<td>String</td>
<td><code>getFileEntry</code></td>
<td>(int <code>index</code>) Get information on the requested file within the zip archive.</td>
</tr>
<tr>
<td>int</td>
<td><code>getFileEntryCount</code></td>
<td>() Get the number of files within the zip archive.</td>
</tr>
<tr>
<td>bool</td>
<td><code>openArchive</code></td>
<td>(string <code>filename</code>, string <code>accessMode=&quot;read&quot;</code>) Open a zip archive for manipulation.</td>
</tr>
<tr>
<td>SimObject</td>
<td><code>openFileForRead</code></td>
<td>(string <code>filename</code>) Open a file within the zip archive for reading.</td>
</tr>
<tr>
<td>SimObject</td>
<td><code>openFileForWrite</code></td>
<td>(string <code>filename</code>) Open a file within the zip archive for writing to.</td>
</tr>
</tbody>
</table>
Detailed Description

Provides access to a zip file.

A ZipObject add, delete and extract files that are within a zip archive. You may also read and write directly to the files within the archive by obtaining a StreamObject for the file.

Example:

```java
// Open a zip archive, creating it if it doesn't exist
%archive = new ZipObject();
%archive.openArchive("testArchive.zip", Write);

// Add a file to the archive with the given name
%archive.addFile("./water.png", "water.png";

// Close the archive to save the changes
%archive.closeArchive();
```

Note:

Behind the scenes all of the work is being done with the ZipArchive and StreamObject classes.

See also:

StreamObject when using methods such as openFileForRead() and openFileForWrite()
Member Function Documentation

```cpp
bool ZipObject::addFile (string filename, string pathInZip, bool replace = true )
```

Add a file to the zip archive.

**Parameters:**

- `filename`: The path and name of the file to add to the zip archive.
- `pathInZip`: The path and name to be given to the file within the zip archive.
- `replace`: If a file already exists within the zip archive at the same location as this new file, this parameter indicates if it should be replaced. By default, it will be replaced.

**Returns:**

True if the file was successfully added to the zip archive.

```cpp
void ZipObject::closeArchive ( )
```

Close an already opened zip archive.

**See also:**

- `openArchive()`

```cpp
void ZipObject::closeFile (SimObject stream )
```

Close a previously opened file within the zip archive.

**Parameters:**
stream The StreamObject of a previously opened file within the zip archive.

See also:
openFileForRead()
openFileForWrite()

bool ZipObject::deleteFile (string pathInZip )

Deleted the given file from the zip archive.

Parameters:

pathInZip The path and name of the file to be deleted from the zip archive.

Returns:
True of the file was successfully deleted.

Note:
Files that have been deleted from the archive will still show up with a getFileEntryCount() until you close the archive. If you need to have the file count up to date with only valid files within the archive, you could close and then open the archive again.

See also:
getFileEntryCount()
closeArchive()
openArchive()

bool ZipObject::extractFile (string pathInZip, string filename)

Extract a file from the zip archive and save it to the requested
location.

**Parameters:**

- pathInZip The path and name of the file to be extracted within the zip archive.
- filename The path and name to give the extracted file.

**Returns:**
True if the file was successfully extracted.

```java
String ZipObject::getFileEntry(int index)
```

Get information on the requested file within the zip archive.

This methods provides five different pieces of information for the requested file:

- filename - The path and name of the file within the zip archive
- uncompressed size
- compressed size
- compression method
- CRC32

Use `getFileNameEntryCount()` to obtain the total number of files within the archive.

**Parameters:**

- index The index of the file within the zip archive. Use `getFileNameEntryCount()` to determine the number of files.

**Returns:**
A tab delimited list of information on the requested file, or an empty string if the file could not be found.

**See also:**
`getFileNameEntryCount()`
int ZipObject::getFileEntryCount()

Get the number of files within the zip archive.

Use `getFileEntry()` to retrieve information on each file within the archive.

**Returns:**

The number of files within the zip archive.

**Note:**

The returned count will include any files that have been deleted from the archive using `deleteFile()`. To clear out all deleted files, you could close and then open the archive again.

**See also:**

`getFileEntry()`, `closeArchive()`, `openArchive()`

---

bool ZipObject::openArchive(string filename,
                             string accessMode = "read")

Open a zip archive for manipulation.

Once a zip archive is opened use the various `ZipObject` methods for working with the files within the archive. Be sure to close the archive when you are done with it.

**Parameters:**

- `filename` The path and file name of the zip archive to open.
- `accessMode` One of read, write or readwrite
Returns:
True is the archive was successfully opened.

Note:
If you wish to make any changes to the archive, be sure to open it with a write or readwrite access mode.

See also:
closeArchive()

SimObject ZipObject::openFileForRead (string filename)

Open a file within the zip archive for reading.
Be sure to close the file when you are done with it.

Parameters:

filename The path and name of the file to open within the zip archive.

Returns:
A standard StreamObject is returned for working with the file.

Note:
You must first open the zip archive before working with files within it.

See also:
closeFile()
oopenArchive()

SimObject ZipObject::openFileForWrite (string filename)

Open a file within the zip archive for writing to.
Be sure to close the file when you are done with it.

**Parameters:**

*filename*  The path and name of the file to open within the zip archive.

**Returns:**

A standard *StreamObject* is returned for working with the file.

**Note:**

You must first open the zip archive before working with files within it.

**See also:**

*closeFile()*

*openArchive()*

---

Copyright © GarageGames, LLC. All Rights Reserved.
Zone Class Reference
[Miscellaneous]

An object that represents an interior space. More...

Inheritance diagram for Zone:

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>void dumpZoneState (bool updateFirst=true)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dump a list of all objects assigned to the zone to the console as well as a list of all connected zone spaces.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>int getZoneId ()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get the unique numeric ID of the zone in its scene.</td>
</tr>
</tbody>
</table>
Public Attributes

Lighting

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorF</td>
<td>ambientLightColor</td>
</tr>
<tr>
<td></td>
<td>Color of ambient lighting in this zone.</td>
</tr>
<tr>
<td>bool</td>
<td>useAmbientLightColor</td>
</tr>
<tr>
<td></td>
<td>Whether to use ambientLightColor for ambient lighting in this zone or the global ambient color.</td>
</tr>
</tbody>
</table>

Internal

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>edge</td>
</tr>
<tr>
<td></td>
<td>For internal use only.</td>
</tr>
<tr>
<td>string</td>
<td>plane</td>
</tr>
<tr>
<td></td>
<td>For internal use only.</td>
</tr>
<tr>
<td>string</td>
<td>point</td>
</tr>
<tr>
<td></td>
<td>For internal use only.</td>
</tr>
</tbody>
</table>

Sound

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SFXAmbience</td>
<td>soundAmbience</td>
</tr>
<tr>
<td></td>
<td>Ambient sound environment for the space.</td>
</tr>
</tbody>
</table>

Zoning

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>zoneGroup</td>
</tr>
<tr>
<td></td>
<td>ID of group the zone is part of.</td>
</tr>
</tbody>
</table>
## Static Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static bool</td>
<td><code>isRenderable</code></td>
<td>Disables rendering of all instances of this type.</td>
</tr>
<tr>
<td>static bool</td>
<td><code>isSelectable</code></td>
<td>Disables selection of all instances of this type.</td>
</tr>
</tbody>
</table>
Detailed Description

An object that represents an interior space.

A zone is an invisible volume that encloses an interior space. All objects that have their world space axis-aligned bounding boxes (AABBs) intersect the zone's volume are assigned to the zone. This assignment happens automatically as objects are placed and transformed. Also, assignment is not exclusive meaning that an object can be assigned to many zones at the same time if it intersects all of them.

In itself, the volume of a zone is fully sealed off from the outside. This means that while viewing the scene from inside the volume, only objects assigned to the zone are rendered while when viewing the scene from outside the volume, objects exclusively only assigned the zone are not rendered.

Usually, you will want to connect zones to each other by means of portals. A portal overlapping with a zone

Example:

```// Example declaration of a Zone. This creates a box-shaped zone.
new Zone( TestZone )
{
    position = "3.61793 -1.01945 14.7442";
    rotation = "1 0 0 0";
    scale = "10 10 10";
};```
Zone Groups

Normally, Zones will not connect to each other when they overlap. This means that if viewing the scene from one zone, the contents of the other zone will not be visible except when there is a portal connecting the zones. However, sometimes it is convenient to represent a single interior space through a combination of Zones so that when any of these zones is visible, all other zones that are part of the same interior space are visible. This is possible by employing "zone groups".

See also:

- Portal
Member Function Documentation

**void Zone::dumpZoneState(bool updateFirst = true)**

Dump a list of all objects assigned to the zone to the console as well as a list of all connected zone spaces.

**Parameters:**

*updateFirst*  
Whether to update the contents of the zone before dumping. Since zoning states of objects are updated on demand, the zone contents can be outdated.

**int Zone::getZoneId( )**

Get the unique numeric ID of the zone in its scene.

**Returns:**

The ID of the zone.
Member Data Documentation

**ColorF** Zone::ambientLightColor

Color of ambient lighting in this zone.
Only used if `useAmbientLightColor` is true.

**string** Zone::edge

For internal use only.

**string** Zone::plane

For internal use only.

**string** Zone::point

For internal use only.

**SFXAmbience** Zone::soundAmbience

Ambient sound environment for the space.

**bool** Zone::useAmbientLightColor

Whether to use `ambientLightColor` for ambient lighting in this zone or the global ambient color.
int Zone::zoneGroup

ID of group the zone is part of.
Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

- BanList
- EditorIconRegistry
- GFXCardProfiler
- GFXCardProfilerAPI
- GFXInit
- GuiArrayCtrl
  - GuiConsole
  - GuiTextListCtrl
  - GuiTreeViewCtrl
- LightningStrikeEvent
- PfxVis
- RadialImpulseEvent
- SimObject
  - ActionMap
  - ArrayObject
  - ConsoleLogger
  - CubemapData
  - DebugDrawer
  - EventManager
  -FileDialog
    - OpenFileDialog
      - OpenFolderDialog
    - SaveFileDialog
  - FileObject
  - ForestBrushElement
  - GFXSamplerStateData
  - GFXStateBlockData
  - GuiControlProfile
    - GuiGameListMenuProfile
      - GuiGameListOptionsProfile
  - GuiCursor
  - LangTable
  - Material
- CustomMaterial
  - Message
  - MessageVector
- NetObject
  - LevelInfo
  - MissionArea
  - SceneObject
    - BasicClouds
    - CloudLayer
    - ConvexShape
    - DecalManager
    - DecalRoad
    - Forest
    - ForestWindEmitter
    - fxFoliageReplicator
    - fxShapeReplicatedStatic
    - fxShapeReplicator
- GameBase
  - Debris
  - Explosion
  - Lightning
  - ParticleEmitter
  - ParticleEmitterNode
  - PhysicsDebris
  - PhysicsShape
  - Precipitation
  - Projectile
  - PxCloth
  - PxMultiActor
- ShapeBase
  - Camera
  - Item
    - ProximityMine
    - TurretShape
    - AI_TurretShape
- MissionMarker
  - SpawnSphere
  - WayPoint
- PathCamera
- Player
  - AIPlayer
  - RigidShape
  - StaticShape
  - Vehicle
    - FlyingVehicle
    - HoverVehicle
    - WheeledVehicle
- Splash
- Trigger
- GroundCover
- GroundPlane
- InteriorInstance
- LightBase
  - PointLight
  - SpotLight
- Marker
- MeshRoad
- OcclusionVolume
- PhysicalZone
- PhysicsForce
- Prefab
- RenderMeshExample
- RenderObjectExample
- RenderShapeExample
- ScatterSky
- SFXEmitter
- SFXSpace
- SkyBox
- Sun
- TerrainBlock
- TimeOfDay
- TSStatic
- WaterObject
  - River
  - WaterBlock
  - WaterPlane
- Zone
  - Portal
- SimpleNetObject
  - RenderBinManager
    - RenderImposterMgr
    - RenderMeshMgr
    - RenderObjectMgr
    - RenderOcclusionMgr
    - RenderPassStateBin
    - RenderTerrainMgr
    - RenderTexTargetBinManager
      - AdvancedLightBinManager
      - RenderGlowMgr
      - RenderParticleMgr
      - RenderPrePassMgr
    - RenderTranslucentMgr
  - RenderPassManager
  - RenderPassStateToken
    - RenderFormatToken
  - ScriptMsgListener
    - MessageForwarder
  - ScriptObject
  - SFXParameter
  - ShaderData
  - SimDataBlock
    - DecalData
    - ForestItemData
      - TSForestItemData
    - GameBaseData
      - DebrisData
      - ExplosionData
      - LightningData
      - ParticleEmitterData
      - ParticleEmitterNodeData
      - PhysicsDebrisData
      - PhysicsShapeData
      - PrecipitationData
      - ProjectileData
      - PxMultiActorData
    - ShapeBaseData
      - CameraData
- ItemData
  - ProximityMineData
  - TurretShapeData
    - AITurretShapeData
- MissionMarkerData
- PathCameraData
- PlayerData
- RigidShapeData
- StaticShapeData
- VehicleData
  - FlyingVehicleData
  - HoverVehicleData
  - WheeledVehicleData
- ShapeBaseImageData
- SplashData
- TriggerData
- LightAnimData
- LightDescription
- LightFlareData
- ParticleData
- PxMaterial
- ReflectorDesc
- SFXAmbience
- SFXDescription
- SFXEnvironment
- SFXFMODEventGroup
- SFXFMODProject
- SFXState
- SFXTrack
  - SFXFMODEvent
  - SFXPlayList
  - SFXProfile
- WheeledVehicleSpring
- WheeledVehicleTire
- SimSet
  - SimGroup
    - GuiControl
      - GuiBitmapBorderCtrl
      - GuiBitmapCtrl
- GuiCrossHairHud
- GuiFadeinBitmapCtrl
- GuiSpeedometerHud
- GuiButtonBaseCtrl
  - GuiBorderButtonCtrl
  - GuiButtonCtrl
    - GuiBitmapButtonCtrl
      - GuiBitmapButtonTextCtrl
    - GuiIconButtonCtrl
    - GuiToggleButtonCtrl
- GuiCheckBoxCtrl
  - GuiRadioCtrl
- GuiSwatchButtonCtrl
- GuiCanvas
- GuiChunkedBitmapCtrl
- GuiClockHud
- GuiContainer
  - GuiFrameSetCtrl
  - GuiPanel
  - GuiScrollCtrl
  - GuiSplitContainer
  - GuiTabBookCtrl
- GuiTextCtrl
  - GuiBubbleTextCtrl
  - GuiPopUpMenuCtrl
  - GuiPopUpMenuCtrlEx
  - GuiProgressBitmapCtrl
  - GuiProgressCtrl
  - GuiTabPageCtrl
  - GuiTextEditCtrl
    - GuiConsoleEditCtrl
    - GuiTextEditSliderBitmapCtrl
    - GuiTextEditSliderCtrl
- GuiTSCtrl
  - GameTSCtrl
  - GuiObjectView
- GuiWindowCtrl
- GuiControlArrayControl
- GuiDragAndDropControl
- GuiDynamicCtrlArrayControl
- GuiGameListMenuCtrl
  - GuiGameListOptionsCtrl
- GuiGraphCtrl
- GuiHealthBarHud
- GuiHealthTextHud
- GuiListBoxCtrl
  - GuiDirectoryFileListCtrl
- GuiMessageVectorCtrl
- GuiMLTextCtrl
  - GuiMLTextEditCtrl
- GuiMouseEventCtrl
  - GuiInputCtrl
- GuiPaneControl
- GuiRolloutCtrl
- GuiScriptNotifyCtrl
- GuiSeparatorCtrl
- GuiShapeNameHud
- GuiSliderCtrl
- GuiStackControl
- GuiTheoraCtrl
- GuiTickCtrl
  - GuiAutoScrollCtrl
  - GuiMenuBar
- NetConnection
  - GameConnection
    - AIConnection
      - AIClient
- Path
- PostEffect
- ScriptGroup
- SFXSource
  - SFXController
  - SFXFMODEventSource
  - SFXSound
- SimXMLDocument
- StreamObject
  - FileStreamObject
- TCPObjec
- HTTPObject
  - TerrainMaterial
  - TSShapeConstructor
  - ZipObject
- SimpleMessageEvent

Copyright © GarageGames, LLC. All Rights Reserved.
Here is a list of all class members with links to the classes they belong to:
- a -

- **aaLevel**: `RenderFormatToken`
- **acceleration**: `SplashData`
- **accelerator**: `GuiControl`
- **acceptsAsChild()**: `SimSet`
- **accuFire**: `ShapeBaseImageData`
- **activate()**: `PhysicalZone, SFXState`
- **activateGhosting()**: `GameConnection`
- **activateRow()**: `GuiGameListMenuCtrl`
- **activateTurret()**: `AITurretShape`
- **active**: `GuiControl, ParticleEmitterNode`
- **add()**: `GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx, SimSet, BanList, ArrayObject`
- **addAbsolute()**: `BanList`
- **addAutoPlot()**: `GuiGraphCtrl`
- **addCategory()**: `GuiPopUpMenuCtrlEx`
- **addChildSelectionByValue**: `GuiTreeViewController`
- **addCollisionDetail()**: `TSShapeConstructor`
- **addColumn()**: `GuiFrameSetCtrl`
- **addComment()**: `SimXMLDocument`
- **addData()**: `SimXMLDocument`
- **addDatum()**: `GuiGraphCtrl`
- **addFile()**: `ZipObject`
- **addFilteredItem()**: `GuiListBoxCtrl`
- **addGuiControl()**: `GuiControl`
- **addHeader()**: `SimXMLDocument`
- **addImposter()**: `TSShapeConstructor`
- **addLanguage()**: `LangTable`
- **addMarker()**: `RenderPassManager`
- **addManager()**: `RenderPassManager`
- **addMenu()**: `GuiMenuBar`
- **addMenuItem()**: `GuiMenuBar`
- **addMesh()**: `TSShapeConstructor`
- **addNewElement()**: `SimXMLDocument`
- **addNode()**: `TSShapeConstructor`
- **addPage()**: `GuiTabBookCtrl`
- **addParameter()**: `SFXSource`
- **addPrimitive()**: `TSShapeConstructor`
- addReference() : Message
- addressModeU : GFXSamplerStateData
- addressModeV : GFXSamplerStateData
- addressModeW : GFXSamplerStateData
- addRow() : GuiFrameSetCtrl, GuiGameListMenuCtrl, GuiGameListOptionsCtrl, GuiTextListCtrl
- addScheme() : GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
- addSelection() : GuiTreeViewCtrl
- addSequence() : TSShapeConstructor
- addSubmenuItem() : GuiMenuBar
- addText() : GuiMLTextCtrl, SimXMLDocument
- addTimeOfDayEvent() : TimeOfDay
- addToIgnoreList() : AITurretShape
- adjustCenter : TSShapeConstructor
- adjustFloor : TSShapeConstructor
- advancedLightmapSupport : LevelInfo
- airAbsorptionHF : SFXEnvironment
- airControl : PlayerData
- alignDirection : ParticleEmitterData
- alignParticles : ParticleEmitterData
- AlignToTerrain : fxShapeReplicator
- allowAllPoses() : Player
- allowColorChars : GuiMLTextCtrl
- allowCrouching() : Player
- allowedMatches : GuiMessageVectorCtrl
- AllowedTerrainSlope : fxFoliageReplicator, fxShapeReplicator
- allowImageStateAnimation : PlayerData
- allowJetJumping() : Player
- allowJumping() : Player
- allowMultipleSelections : GuiListBoxCtrl
- AllowOnInteriors : fxFoliageReplicator, fxShapeReplicator
- AllowOnStatics : fxFoliageReplicator, fxShapeReplicator
- AllowOnTerrain : fxFoliageReplicator, fxShapeReplicator
- AllowOnWater : fxFoliageReplicator, fxShapeReplicator
- allowPlayerStep : fxShapeReplicatedStatic, TSStatic
- allowProne() : Player
- allowReflectPass : PostEffect
- allowReorder : GuiTabBookCtrl
- `allowSprinting()`: Player
- `allowSwimming()`: Player
- `AllowWaterSurface`: `fxFoliageReplicator`, `fxShapeReplicator`
- `alphaArg1`: `GFXSamplerStateData`
- `alphaArg2`: `GFXSamplerStateData`
- `alphaArg3`: `GFXSamplerStateData`
- `AlphaCutoff`: `fxFoliageReplicator`
- `alphaDefined`: `GFXStateBlockData`
- `alphaRef`: `GFXSamplerStateData`
- `alphaTest`: `Material`
- `alphaTestEnable`: `GFXStateBlockData`
- `alphaTestFunc`: `GFXStateBlockData`
- `alphaTestRef`: `GFXStateBlockData`
- `altCommand`: `GuiControl`
- `alwaysHandleMouseButtons`: `GuiCanvas`
- `alwaysImport`: `TSShapeConstructor`
- `alwaysImportMesh`: `TSShapeConstructor`
- `ambient`: `Sun`
- `ambientFactor`: `ParticleEmitterData`
- `ambientLightBlendCurve`: `LevelInfo`
- `ambientLightBlendPhase`: `LevelInfo`
- `ambientLightColor`: `Zone`
- `ambientScale`: `ScatterSky`
- `anchorBottom`: `GuiContainer`
- `anchorLeft`: `GuiContainer`
- `anchorRight`: `GuiContainer`
- `anchorTop`: `GuiContainer`
- `angularDamping`: `PhysicsDebrisData`, `PhysicsShapeData`
- `angularDrag`: `Camera`, `PxMultiActorData`
- `angularForce`: `Camera`
- `angularSleepThreshold`: `PhysicsDebrisData`, `PhysicsShapeData`
- `animate()`: `TimeOfDay`, `LightBase`, `Sun`
- `animateAllShapes`: `ShapeBaseImageData`
- `animateOnServer`: `ShapeBaseImageData`
- `animateSplashes`: `Precipitation`
- `animateTexture`: `ParticleData`
- `animationDone()`: `PlayerData`
- animationPeriod : LightDescription, LightBase
- animationPhase : LightDescription, LightBase
- animationType : LightBase, LightDescription
- animFlags : Material
- animSequence : GuiObjectView
- animTexFrames : ParticleData
- animTexName : ParticleData
- animTexTiling : ParticleData
- antiSwayForce : WheeledVehicleSpring
- append() : ArrayObject
- appliedForce : PhysicalZone
- apply() : LightFlareData, LightDescription, Sun
- applyChanges() : ScatterSky
- applyDamage() : Lightning, ShapeBase
- applyImpulse() : ShapeBase, GameBase
- applyRadialImpulse() : GameBase
- applyRepair() : ShapeBase
- area : MissionArea
- armingDelay : ProximityMineData, ProjectileData
- armingSound : ProximityMineData
- assignFieldsFrom() : SimObject
- assignPersistentId() : SimObject
- attach() : GuiWindowCtrl, PhysicsForce, GuiMessageVectorCtrl, ConsoleLogger
- attachments : PxCloth
- attachTo() : GuiWindowCtrl
- attachToObject() : ForestWindEmitter
- attenuationRatio : Sun, LightBase, LightDescription, ScatterSky
- attribute() : SimXMLDocument
- attributeExists() : SimXMLDocument
- attributeF32() : SimXMLDocument
- attributeS32() : SimXMLDocument
- autoAngularForce : FlyingVehicleData
- autoBalance : GuiFrameSetCtrl
- autoCellSize : GuiDynamicCtrlArrayControl
- autoCollapseSiblings : GuiRolloutCtrl
- autoFitExtents : GuiBitmapButtonCtrl
- autoFitRadius() : Camera
- autoInputDamping: FlyingVehicleData
- autoLinearForce: FlyingVehicleData
- autoSize: GuiIconButtonCtrl
- autoSizeHeight: GuiControlProfile
- autoSizeWidth: GuiControlProfile
- autoSpawn: SpawnSphere
- autoTriggerDelay: ProximityMineData
- axisTilt: TimeOfDay
- azimuth: ScatterSky, Sun
- azimuthOverride: TimeOfDay
## Namespace List

Here is a list of all namespaces with brief descriptions:

<table>
<thead>
<tr>
<th>Namespace</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$AL</td>
<td></td>
</tr>
<tr>
<td>$BasicLightManager</td>
<td></td>
</tr>
<tr>
<td>$BasicLightManagerStats</td>
<td></td>
</tr>
<tr>
<td>$Con</td>
<td></td>
</tr>
<tr>
<td>$Decals</td>
<td></td>
</tr>
<tr>
<td>$Ease</td>
<td></td>
</tr>
<tr>
<td>$gfx</td>
<td></td>
</tr>
<tr>
<td>$Interior</td>
<td></td>
</tr>
<tr>
<td>$Light</td>
<td></td>
</tr>
<tr>
<td>$Physics</td>
<td></td>
</tr>
<tr>
<td>$platform</td>
<td></td>
</tr>
<tr>
<td>$pref</td>
<td></td>
</tr>
<tr>
<td>$pref::Camera</td>
<td></td>
</tr>
<tr>
<td>$pref::Decals</td>
<td></td>
</tr>
<tr>
<td>$pref::GroundCover</td>
<td></td>
</tr>
<tr>
<td>$pref::imposter</td>
<td></td>
</tr>
<tr>
<td>$pref::Input</td>
<td></td>
</tr>
<tr>
<td>$pref::Interior</td>
<td></td>
</tr>
<tr>
<td>$pref::Net</td>
<td></td>
</tr>
<tr>
<td>$pref::Physics</td>
<td></td>
</tr>
<tr>
<td>$pref::PhysicsDebris</td>
<td></td>
</tr>
<tr>
<td>$pref::ProjectedShadow</td>
<td></td>
</tr>
<tr>
<td>$pref::PSSM</td>
<td></td>
</tr>
<tr>
<td>$pref::Reflect</td>
<td></td>
</tr>
<tr>
<td>$pref::SFX</td>
<td></td>
</tr>
<tr>
<td>$pref::SFX::FMOD</td>
<td></td>
</tr>
<tr>
<td>$pref::Shadows</td>
<td></td>
</tr>
<tr>
<td>$pref::shadows</td>
<td></td>
</tr>
<tr>
<td>$pref::Terrain</td>
<td></td>
</tr>
<tr>
<td>$pref::TS</td>
<td></td>
</tr>
<tr>
<td>$pref::Video</td>
<td></td>
</tr>
<tr>
<td>$pref::Water</td>
<td></td>
</tr>
<tr>
<td>$pref::WorldEditor</td>
<td></td>
</tr>
<tr>
<td>$ProxMine</td>
<td></td>
</tr>
<tr>
<td>$Sampler</td>
<td></td>
</tr>
<tr>
<td>$SB</td>
<td></td>
</tr>
<tr>
<td>$Scene</td>
<td></td>
</tr>
<tr>
<td>$SFX</td>
<td></td>
</tr>
<tr>
<td>$SFX::Device</td>
<td></td>
</tr>
<tr>
<td>$Shadows</td>
<td></td>
</tr>
<tr>
<td>$Stats</td>
<td></td>
</tr>
<tr>
<td>$TSCcontrol</td>
<td></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
$AL Namespace Reference
### Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>PSSMDebugRender</td>
<td>Enables debug rendering of the PSSM shadows.</td>
</tr>
<tr>
<td>bool</td>
<td>UseSSAOMask</td>
<td>Used by the SSAO PostEffect to toggle the sampling of ssaomask texture by the light shaders.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
$BasicLightManager Namespace Reference
Variables

float `shadowFilterDistance`
The maximum distance in meters that projected shadows will get soft filtering.
$BasicLightManagerStats Namespace
Reference
### Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><code>activePlugins</code></td>
<td>The number of active Basic Lighting SceneObjectLightingPlugin objects this frame.</td>
</tr>
<tr>
<td>int</td>
<td><code>elapsedUpdateMs</code></td>
<td>The number of milliseconds spent this frame updating Basic Lighting shadows.</td>
</tr>
<tr>
<td>int</td>
<td><code>shadowsUpdated</code></td>
<td>The number of Basic Lighting shadows updated this frame.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
$Con Namespace Reference
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>alwaysUseDebugOutput</td>
<td>Determines whether to send output to the platform's &quot;debug&quot; system.</td>
</tr>
<tr>
<td>string</td>
<td>File</td>
<td>The currently executing script file.</td>
</tr>
<tr>
<td>bool</td>
<td>logBufferEnabled</td>
<td>If true, the log buffer will be enabled.</td>
</tr>
<tr>
<td>int</td>
<td>objectCopyFailures</td>
<td>If greater than zero then it counts the number of object creation failures based on a missing copy object and does not report an error.</td>
</tr>
<tr>
<td>int</td>
<td>printLevel</td>
<td>This is deprecated.</td>
</tr>
<tr>
<td>string</td>
<td>Root</td>
<td>The mod folder for the currently executing script file.</td>
</tr>
<tr>
<td>bool</td>
<td>useTimestamp</td>
<td>If true, a timestamp is prepended to every console message.</td>
</tr>
<tr>
<td>bool</td>
<td>warnUndefinedVariables</td>
<td>If true, a warning will be displayed in the console whenever a undefined variable is used in script.</td>
</tr>
</tbody>
</table>
$Decals Namespace Reference
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>debugRender</code></td>
<td>If true, the decal spheres will be visualized when in the editor.</td>
</tr>
<tr>
<td>bool</td>
<td><code>poolBuffers</code></td>
<td>If true, will merge all PrimitiveBuffers and VertexBuffers into a pair of pools before clearing them at the end of a frame.</td>
</tr>
<tr>
<td>float</td>
<td><code>sphereDistanceTolerance</code></td>
<td>The distance at which the decal system will start breaking up decal spheres when adding new decals.</td>
</tr>
<tr>
<td>float</td>
<td><code>sphereRadiusTolerance</code></td>
<td>The radius beyond which the decal system will start breaking up decal spheres when adding new decals.</td>
</tr>
</tbody>
</table>
$Ease Namespace Reference
<table>
<thead>
<tr>
<th>int</th>
<th>Variables</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>Back</td>
<td>Backwards ease for curve movement.</td>
</tr>
<tr>
<td>int</td>
<td>Bounce</td>
<td>Bounce ease for curve movement.</td>
</tr>
<tr>
<td>int</td>
<td>Circular</td>
<td>Circular ease for curve movement.</td>
</tr>
<tr>
<td>int</td>
<td>Cubic</td>
<td>Cubic ease for curve movement.</td>
</tr>
<tr>
<td>int</td>
<td>Elastic</td>
<td>Elastic ease for curve movement.</td>
</tr>
<tr>
<td>int</td>
<td>Exponential</td>
<td>Exponential ease for curve movement.</td>
</tr>
<tr>
<td>int</td>
<td>In</td>
<td>In ease for curve movement.</td>
</tr>
<tr>
<td>int</td>
<td>InOut</td>
<td>InOut ease for curve movement.</td>
</tr>
<tr>
<td>int</td>
<td>Linear</td>
<td>Linear ease for curve movement.</td>
</tr>
<tr>
<td>int</td>
<td>Out</td>
<td>Out ease for curve movement.</td>
</tr>
<tr>
<td>int</td>
<td>Quadratic</td>
<td>Quadratic ease for curve movement.</td>
</tr>
<tr>
<td>int</td>
<td>Quartic</td>
<td>Quartic ease for curve movement.</td>
</tr>
<tr>
<td>int</td>
<td>Quintic</td>
<td>Quintic ease for curve movement.</td>
</tr>
<tr>
<td>int</td>
<td>Sinusoidal</td>
<td>Sinusoidal ease for curve movement.</td>
</tr>
</tbody>
</table>
$gfx Namespace Reference
### Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>disableOcclusionQuery</td>
<td>Debug helper that disables all hardware occlusion queries causing them to return only the visible state.</td>
</tr>
<tr>
<td>bool</td>
<td>disassembleAllShaders</td>
<td>On supported devices this will dump shader disassembly to the procedural shader folder.</td>
</tr>
<tr>
<td>bool</td>
<td>wireframe</td>
<td>Used to toggle wireframe rendering at runtime.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
$Interior Namespace Reference
<table>
<thead>
<tr>
<th>bool</th>
<th><strong>DontRestrictOutside</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Render only the outside zone of all InteriorInstances.</td>
</tr>
</tbody>
</table>
Variable Documentation

bool $Interior::DontRestrictOutside

Render only the outside zone of all InteriorInstances.
$Light Namespace Reference
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bool</code></td>
<td><code>renderLightFrustums</code></td>
<td>Toggles rendering of light frustums when the light is selected in the editor.</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><code>renderViz</code></td>
<td>Toggles visualization of light object's radius or cone.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
$Physics Namespace Reference
**Variables**

| bool  isSinglePlayer | Informs the physics simulation if only a single player exists. |

---

Copyright © GarageGames, LLC. All Rights Reserved.
$platform Namespace Reference
### Variables

<table>
<thead>
<tr>
<th>int</th>
<th><code>backgroundSleepTime</code></th>
<th>Controls processor time usage when the game window is out of focus.</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><code>timeManagerProcessInterval</code></td>
<td>Controls processor time usage when the game window is in focus.</td>
</tr>
</tbody>
</table>
$pref Namespace Reference
### Namespaces

<table>
<thead>
<tr>
<th>Namespace</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera</td>
</tr>
<tr>
<td>Decals</td>
</tr>
<tr>
<td>GroundCover</td>
</tr>
<tr>
<td>imposter</td>
</tr>
<tr>
<td>Input</td>
</tr>
<tr>
<td>Interior</td>
</tr>
<tr>
<td>Net</td>
</tr>
<tr>
<td>Physics</td>
</tr>
<tr>
<td>PhysicsDebris</td>
</tr>
<tr>
<td>ProjectedShadow</td>
</tr>
<tr>
<td>PSSM</td>
</tr>
<tr>
<td>Reflect</td>
</tr>
<tr>
<td>SFX</td>
</tr>
<tr>
<td>Shadows</td>
</tr>
<tr>
<td>shadows</td>
</tr>
<tr>
<td>Terrain</td>
</tr>
<tr>
<td>TS</td>
</tr>
<tr>
<td>Video</td>
</tr>
<tr>
<td>Water</td>
</tr>
<tr>
<td>WorldEditor</td>
</tr>
</tbody>
</table>
Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>enableBadWordFilter</code></td>
<td>If true, the bad word filter will be enabled.</td>
</tr>
<tr>
<td>bool</td>
<td><code>enablePostEffects</code></td>
<td>If true, post effects will be enabled.</td>
</tr>
<tr>
<td>float</td>
<td><code>windEffectRadius</code></td>
<td>Radius to affect the wind.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
$pref::Camera
$pref::Camera Namespace Reference
Variables

float distanceScale
   A scale to apply to the normal visible distance, typically used for tuning performance.

Copyright © GarageGames, LLC. All Rights Reserved.
$pref::Decals
$pref::Decals Namespace Reference
### Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>enabled</td>
<td>Controls whether decals are rendered.</td>
</tr>
<tr>
<td>float</td>
<td>lifeTimeScale</td>
<td>Lifetime that decals will last after being created in the world. Deprecated. Use DecalData::lifeSpan instead.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
$pref::GroundCover
$pref::GroundCover Namespace Reference
Variables

float densityScale
A global LOD scalar which can reduce the overall density of placed GroundCover.
$pref::imposter
$pref::imposter Namespace Reference
Variables

bool canShadow

User preference which toggles shadows from imposters. Defaults to true.

Copyright © GarageGames, LLC. All Rights Reserved.
$pref::Input
$pref::Input Namespace Reference
## Variables

<table>
<thead>
<tr>
<th>bool</th>
<th>JoystickEnabled</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If true, the joystick is currently enabled.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
$pref::Interior
$pref::Interior Namespace Reference
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>detailAdjust</td>
<td>For all InteriorInstance rendering to a particular detail level.</td>
</tr>
<tr>
<td>bool</td>
<td>VertexLighting</td>
<td>Forces all InteriorInstances to not render their lightmaps.</td>
</tr>
</tbody>
</table>
Variable Documentation

float $pref::Interior::detailAdjust

Forces all InteriorInstance rendering to a particular detail level.

bool $pref::Interior::VertexLighting

Forces all InteriorInstances to not render their lightmaps.
$pref::Net
$pref::Net Namespace Reference
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>LagThreshold</td>
<td>How long between received packets before the client is considered as lagging (in ms).</td>
</tr>
<tr>
<td>int</td>
<td>PacketRateToClient</td>
<td>Sets how often packets are sent from the server to a client.</td>
</tr>
<tr>
<td>int</td>
<td>PacketRateToServer</td>
<td>Sets how often packets are sent from the client to the server.</td>
</tr>
<tr>
<td>int</td>
<td>PacketSize</td>
<td>Sets the maximum size in bytes an individual network packet may be.</td>
</tr>
</tbody>
</table>
$pref::Physics
$pref::Physics Namespace Reference
## Variables

<table>
<thead>
<tr>
<th>int threadCount</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of threads to use in a single pass of the physics engine.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
$pref::PhysicsDebris
$pref::PhysicsDebris Namespace Reference
## Variables

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>lifetimeScale</code></td>
<td>Scales how long PhysicsDebris will live before being removed.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
$pref::ProjectedShadow
$pref::ProjectedShadow Namespace
Reference
### Variables

<table>
<thead>
<tr>
<th>float</th>
<th>fadeEndPixelSize</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A size in pixels at which BL shadows are fully faded out. This should be a smaller value than fadeStartPixelSize.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>float</th>
<th>fadeStartPixelSize</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A size in pixels at which BL shadows begin to fade out. This should be a larger value than fadeEndPixelSize.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
$pref::PSSM
$pref::PSSM Namespace Reference
### Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>detailAdjustScale</td>
<td>Scales the model LOD when rendering into the <strong>PSSM</strong> shadow. Use this to reduce the draw calls when rendering the shadow by having meshes LOD out nearer to the camera than normal.</td>
</tr>
<tr>
<td>float</td>
<td>smallestVisiblePixelSize</td>
<td>The smallest pixel size an object can be and still be rendered into the <strong>PSSM</strong> shadow. Use this to force culling of small objects which contribute little to the final shadow.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
$pref::Reflect
$pref::Reflect Namespace Reference
### Variables

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><code>frameLimitMS</code></td>
<td>ReflectionManager tries not to spend more than this amount of time updating reflections per frame.</td>
</tr>
<tr>
<td>float</td>
<td><code>refractTexScale</code></td>
<td>RefractTex has dimensions equal to the active render target scaled in both x and y by this float.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
$pref::SFX
$pref::SFX Namespace Reference
Namespaces

```cpp
namespace FMOD
```

Copyright © GarageGames, LLC. All Rights Reserved.
$pref::SFX::FMOD
$pref::SFX::FMOD Namespace Reference
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>disableSoftware</code></td>
<td>Whether to disable the FMOD software mixer to conserve memory.</td>
</tr>
<tr>
<td>string</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>DSoundHRTF</code></td>
<td>The type of HRTF to use for hardware-mixed 3D sounds when FMOD is using DirectSound for sound output and hardware-acceleration is not available.</td>
</tr>
<tr>
<td>bool</td>
<td><code>enableProfile</code></td>
<td>Whether to enable support for FMOD's profiler.</td>
</tr>
<tr>
<td>string</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>pluginPath</code></td>
<td>Path to additional FMOD plugins.</td>
</tr>
<tr>
<td>bool</td>
<td><code>useSoftwareHRTF</code></td>
<td>Whether to enable HRTF in FMOD's software mixer.</td>
</tr>
<tr>
<td>bool</td>
<td><code>useSoftwareReverbLowmem</code></td>
<td>If true, FMOD's SFX reverb is run using 22/24kHz delay buffers, halving the memory required.</td>
</tr>
</tbody>
</table>
$pref::shadows
$pref::shadows Namespace Reference
Variables

<table>
<thead>
<tr>
<th>ShadowFilterMode</th>
<th>filterMode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The filter mode to use for shadows.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
$pref::Terrain
$pref::Terrain Namespace Reference
### Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>detailScale</code></td>
<td>A global detail scale used to tweak the material detail distances.</td>
</tr>
<tr>
<td>float</td>
<td><code>lodScale</code></td>
<td>A global LOD scale used to tweak the default terrain screen error value.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
$pref::TS
$pref::TS Namespace Reference
## Variables

**float** `detailAdjust`  
User preference for scaling the TSShape level of detail. The smaller the value the closer the camera must get to see the highest LOD. This setting can have a huge impact on performance in mesh heavy scenes. The default value is 1.

**int** `maxInstancingVerts`  
Enables mesh instancing on non-skin meshes that have less that this count of verts. The default value is 200. Higher values can degrade performance.

**int** `skipLoadDLs`  
User preference which causes TSShapes to skip loading higher lods. This potentially reduces the GPU resources and materials generated as well as limits the LODs rendered. The default value is 0.

**int** `skipRenderDLs`  
User preference which causes TSShapes to skip rendering higher lods. This will reduce the number of draw calls and triangles rendered and improve rendering performance when proper LODs have been created for your models. The default value is 0.

**float** `smallestVisiblePixelSize`  
User preference which sets the smallest pixel size at which TSShapes will skip rendering. This will force all shapes to stop rendering when they get smaller than this size. The default value is -1 which disables it.
$pref::Video
$pref::Video Namespace Reference
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><code>defaultAnisotropy</code></td>
<td>Global variable defining the default anisotropy value.</td>
</tr>
<tr>
<td>bool</td>
<td><code>disableVerticalSync</code></td>
<td>Disables vertical sync on the active device.</td>
</tr>
<tr>
<td>float</td>
<td><code>forcedPixVersion</code></td>
<td>Will force the shader model if the value is positive and less than the shader model supported by the active device. Use 0 for fixed function.</td>
</tr>
<tr>
<td>int</td>
<td><code>textureReductionLevel</code></td>
<td>The number of mipmap levels to drop on loaded textures to reduce video memory usage. It will skip any textures that have been defined as not allowing down scaling.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
$pref::Water
$pref::Water Namespace Reference
Variables

**bool disableTrueReflections**
Force all water objects to use static cubemap reflections.

Copyright © GarageGames, LLC. All Rights Reserved.
$pref::WorldEditor
$pref::WorldEditor Namespace Reference
### Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td><code>cameraFOV</code></td>
<td>Field of view for editor's perspective camera, in degrees.</td>
</tr>
<tr>
<td>float</td>
<td><code>visibleDistanceScale</code></td>
<td>Scale factor for the visible render distance.</td>
</tr>
</tbody>
</table>
## Variable Documentation

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>float $pref::WorldEditor::cameraFOV</code></td>
<td>Field of view for editor's perspective camera, in degrees.</td>
</tr>
<tr>
<td><code>float $pref::WorldEditor::visibleDistanceScale</code></td>
<td>Scale factor for the visible render distance.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
$ProxMine Namespace Reference
### Variables

<table>
<thead>
<tr>
<th>int</th>
<th><code>autoDeleteTicks</code></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number of ticks until an exploded mine is deleted on the server.</td>
</tr>
</tbody>
</table>
Variable Documentation

| int $ProxMine::autoDeleteTicks |

Number of ticks until an exploded mine is deleted on the server.

After a mine has exploded it remains in the server's scene graph for a time to allow its exploded state to be passed along to each client. This variable controls how long a mine remains before it is deleted. Any client that has not received the exploded state by then (perhaps due to lag) will not see any explosion produced by the mine.
$Sampler Namespace Reference
Variables

int frequency
Samples taken every nth frame.
$SB Namespace Reference
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>CloakSpeed</td>
<td>Time to cloak, in seconds.</td>
</tr>
<tr>
<td>float</td>
<td>DFDec</td>
<td>Speed to reduce the damage flash effect per tick.</td>
</tr>
<tr>
<td>float</td>
<td>FullCorrectionDistance</td>
<td>Distance at which a weapon's muzzle vector is fully corrected to match where the player is looking.</td>
</tr>
<tr>
<td>float</td>
<td>WODec</td>
<td>Speed to reduce the whiteout effect per tick.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
$Scene Namespace Reference
### Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>disableTerrainOcclusion</td>
<td>Used to disable the somewhat expensive terrain occlusion testing.</td>
</tr>
<tr>
<td>bool</td>
<td>disableZoneCulling</td>
<td>If true, zone culling will be disabled and the scene contents will only be culled against the root frustum.</td>
</tr>
<tr>
<td>bool</td>
<td>lockCull</td>
<td>Debug tool which locks the frustum culling to the current camera location.</td>
</tr>
<tr>
<td>int</td>
<td>maxOccludersPerZone</td>
<td>Maximum number of occluders that will be concurrently allowed into the scene culling state of any given zone.</td>
</tr>
<tr>
<td>float</td>
<td>occluderMinHeightPercentage</td>
<td>TODO.</td>
</tr>
<tr>
<td>float</td>
<td>occluderMinWidthPercentage</td>
<td>TODO.</td>
</tr>
<tr>
<td>bool</td>
<td>renderBoundingBoxes</td>
<td>If true, the bounding boxes of objects will be displayed.</td>
</tr>
</tbody>
</table>
$SFX Namespace Reference
Namespaces

namespace Device
# Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>ambientUpdateTime</td>
<td>Milliseconds spent on the last ambient audio update.</td>
</tr>
<tr>
<td>const int</td>
<td>DEVICE_CAPS_DSPEFFECTS</td>
<td>Sound device capability flag indicating that the sound device supports adding DSP effect chains to sounds.</td>
</tr>
<tr>
<td>const int</td>
<td>DEVICE_CAPS_FMODDESIGNER</td>
<td>Sound device capability flag indicating that the sound device supports FMOD Designer audio projects.</td>
</tr>
<tr>
<td>const int</td>
<td>DEVICE_CAPS_MULTILISTENER</td>
<td>Sound device capability flag indicating that the sound device supports multiple concurrent listeners.</td>
</tr>
<tr>
<td>const int</td>
<td>DEVICE_CAPS_OCCLUSION</td>
<td>Sound device capability flag indicating that the sound device implements sound occlusion.</td>
</tr>
<tr>
<td>const int</td>
<td>DEVICE_CAPS_REVERB</td>
<td>Sound device capability flag indicating that the sound device supports reverb.</td>
</tr>
<tr>
<td>const int</td>
<td>DEVICE_CAPS_VOICEMANAGEMENT</td>
<td>Sound device capability flag indicating that the sound device implements its own voice virtualization.</td>
</tr>
<tr>
<td>const int</td>
<td>DEVICE_INFO_CAPS</td>
<td>Index of device capability flags in device info string.</td>
</tr>
<tr>
<td>const int</td>
<td>DEVICE_INFO_MAXBUFFERS</td>
<td>Index of buffer limit number in device info string.</td>
</tr>
<tr>
<td>const int</td>
<td>DEVICE_INFO_NAME</td>
<td>Index of device name field in device info string.</td>
</tr>
<tr>
<td>const int</td>
<td>DEVICE_INFO_PROVIDER</td>
<td>Index of sound provider field in device info string.</td>
</tr>
<tr>
<td>const int</td>
<td>DEVICE_INFO_USEHARDWARE</td>
<td></td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>int <code>numCulled</code></td>
<td>Number of SFXSounds that are currently in virtualized playback mode.</td>
<td></td>
</tr>
<tr>
<td>int <code>numPlaying</code></td>
<td>Number of SFXSources that are currently in playing state.</td>
<td></td>
</tr>
<tr>
<td>int <code>numSounds</code></td>
<td>Number of <code>SFXSound</code> type objects (i.e. actual single-file sounds) that are currently instantiated.</td>
<td></td>
</tr>
<tr>
<td>int <code>numSources</code></td>
<td>Number of <code>SFXSource</code> type objects that are currently instantiated.</td>
<td></td>
</tr>
<tr>
<td>int <code>numVoices</code></td>
<td>Number of voices that are currently allocated on the sound device.</td>
<td></td>
</tr>
<tr>
<td>int <code>parameterUpdateTime</code></td>
<td>Milliseconds spent on the last <code>SFXParameter</code> update loop.</td>
<td></td>
</tr>
<tr>
<td>int <code>sourceUpdateTime</code></td>
<td>Milliseconds spent on the last <code>SFXSource</code> update loop.</td>
<td></td>
</tr>
</tbody>
</table>
$SFX::Device
$SFX::Device Namespace Reference
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>fmodCoreMem</td>
<td>Current number of bytes allocated by the core FMOD sound system.</td>
</tr>
<tr>
<td>int</td>
<td>fmodEventMem</td>
<td>Current number of bytes allocated by the FMOD Designer event system.</td>
</tr>
<tr>
<td>int</td>
<td>fmodNumEventSources</td>
<td>The current number of SFXFMODEventSource instances in the system.</td>
</tr>
</tbody>
</table>
$Shadows Namespace Reference
Variables

bool disable

Used by the editor to disable all shadow rendering.

Copyright © GarageGames, LLC. All Rights Reserved.
<table>
<thead>
<tr>
<th>Variables</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int netBitsReceived</td>
<td>The number of bytes received during the last packet process operation.</td>
</tr>
<tr>
<td>int netBitsSent</td>
<td>The number of bytes sent during the last packet send operation.</td>
</tr>
<tr>
<td>int netGhostUpdates</td>
<td>The total number of ghosts added, removed, and/or updated on the client during the last packet process operation.</td>
</tr>
</tbody>
</table>
$TSControl Namespace Reference
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>frameCount</td>
<td>The number of frames that have been rendered since this control was created.</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
Here is a list of all namespace members with links to the namespace documentation for each member:

- a -
  - activePlugins : $BasicLightManagerStats
  - alwaysUseDebugOutput : $Con
  - ambientUpdateTime : $SFX
  - autoDeleteTicks : $ProxMine
- b -
  - Back : $Ease
  - backgroundSleepTime : $platform
  - Bounce : $Ease

- c -
  - cameraFOV : $pref::WorldEditor
  - canShadow : $pref::imposter
  - Circular : $Ease
  - CloakSpeed : $SB
  - Cubic : $Ease

- d -
  - debugRender : $Decals
  - defaultAnisotropy : $pref::Video
  - densityScale : $pref::GroundCover
  - detailAdjust : $pref::TS , $pref::Interior
  - detailAdjustScale : $pref::PSSM
  - detailScale : $pref::Terrain
  - DEVICE_CAPS_DSPEFFECTS : $SFX
  - DEVICE_CAPS_FMODDESIGNER : $SFX
  - DEVICE_CAPS_MULTILISTENER : $SFX
  - DEVICE_CAPS_OCCLUSION : $SFX
  - DEVICE_CAPS_REVERB : $SFX
  - DEVICE_CAPS_VOICEMANAGEMENT : $SFX
  - DEVICE_INFO_CAPS : $SFX
  - DEVICE_INFO_MAXBUFFERS : $SFX
  - DEVICE_INFO_NAME : $SFX
  - DEVICE_INFO_PROVIDER : $SFX
  - DEVICE_INFO_USEHARDWARE : $SFX
  - DFDec : $SB
  - disable : $pref::Shadows , $Shadows
  - disableOcclusionQuery : $gfx
  - disableSoftware : $pref::SFX::FMOD
  - disableTerrainOcclusion : $Scene
  - disableTrueReflections : $pref::Water
- e -
  - elapsedUpdateMs : $BasicLightManagerStats
  - Elastic : $Ease
  - enableBadWordFilter : $pref
  - enabled : $pref::Decals
  - enablePostEffects : $pref
  - enableProfile : $pref::SFX::FMOD
  - Exponential : $Ease

- f -
  - fadeEndPixelSize : $pref::ProjectedShadow
  - fadeStartPixelSize : $pref::ProjectedShadow
  - File : $Con
  - filterMode : $pref::shadows
  - fmodCoreMem : $SFX::Device
  - fmodEventMem : $SFX::Device
  - fmodNumEventSources : $SFX::Device
  - forcedPixVersion : $pref::Video
  - frameCount : $TSControl
  - frameLimitMS : $pref::Reflect
  - frequency : $Sampler
  - FullCorrectionDistance : $SB

- i -
  - In : $Ease
  - InOut : $Ease
  - isSinglePlayer : $Physics

- j -
- L -
  - JoystickEnabled : $pref::Input

- m -
  - LagThreshold : $pref::Net
  - lifetimeScale : $pref::PhysicsDebris
  - lifeTimeScale : $pref::Decals
  - Linear : $Ease
  - lockCull : $Scene
  - lodScale : $pref::Terrain
  - logBufferEnabled : $Con

- n -
  - maxInstancingVerts : $pref::TS
  - maxOccludersPerZone : $Scene

- o -
  - objectCopyFailures : $Con
  - occluderMinHeightPercentage : $Scene
  - occluderMinWidthPercentage : $Scene
  - Out : $Ease

- p -
  - PacketRateToClient : $pref::Net
  - PacketRateToServer : $pref::Net
  - PacketSize : $pref::Net
- p -
  - parameterUpdateTime : $SFX
  - pluginPath : $pref::SFX::FMOD
  - poolBuffers : $Decals
  - printLevel : $Con
  - PSSMDebugRender : $AL

- q -
  - Quadratic : $Ease
  - Quartic : $Ease
  - Quintic : $Ease

- r -
  - refractTexScale : $pref::Reflect
  - renderBoundingBoxes : $Scene
  - renderLightFrustums : $Light
  - renderViz : $Light
  - Root : $Con

- s -
  - shadowFilterDistance : $BasicLightManager
  - shadowsUpdated : $BasicLightManagerStats
  - Sinusoidal : $Ease
  - skipLoadDLs : $pref::TS
  - skipRenderDLs : $pref::TS
  - smallestVisiblePixelSize : $pref::PSSM , $pref::TS
  - sourceUpdateTime : $SFX
  - sphereDistanceTolerance : $Decals
  - sphereRadiusTolerance : $Decals

- t -
  - textureReductionLevel : $pref::Video
  - textureScalar : $pref::Shadows
  - threadCount : $pref::Physics
  - timeManagerProcessInterval : $platform

- u -
- v -

- useSoftwareHRTF : $pref::SFX::FMOD
- useSoftwareReverbLowmem : $pref::SFX::FMOD
- UseSSAOMask : $AL
- useTimestamp : $Con

- w -

- VertexLighting : $pref::Interior
- visibleDistanceScale : $pref::WorldEditor

- warnUndefinedVariables : $Con
- windEffectRadius : $pref
- wireframe : $gfx
- WODec : $SB

Copyright © GarageGames, LLC. All Rights Reserved.
Graph Legend

This page explains how to interpret the graphs that are generated by doxygen.

Consider the following example:

```cpp
/*! Invisible class because of truncation */
class Invisible { };

/*! Truncated class, inheritance relation is hidden */
class Truncated : public Invisible { };

/*! Class not documented with doxygen comments */
class Undocumented { };

/*! Class that is inherited using public inheritance */
class PublicBase : public Truncated { };

/*! A template class */
template<class T> class Templ { };

/*! Class that is inherited using protected inheritance */
class ProtectedBase { };

/*! Class that is inherited using private inheritance */
class PrivateBase { };

/*! Class that is used by the Inherited class */
class Used { };

/*! Super class that inherits a number of other classes */
class Inherited : public PublicBase,
```
protected ProtectedBase,
private PrivateBase,
public Undocumented,
public Templ<int>
{
    private:
        Used *m_usedClass;
};

This will result in the following graph:

The boxes in the above graph have the following meaning:

- A filled gray box represents the struct or class for which the graph is generated.
- A box with a black border denotes a documented struct or class.
- A box with a grey border denotes an undocumented struct or class.
- A box with a red border denotes a documented struct or class for which not all inheritance/containment relations are shown. A graph is truncated if it does not fit within the specified boundaries.

The arrows have the following meaning:

- A dark blue arrow is used to visualize a public inheritance relation between two classes.
- A dark green arrow is used for protected inheritance.
• A dark red arrow is used for private inheritance.
• A purple dashed arrow is used if a class is contained or used by another class. The arrow is labeled with the variable(s) through which the pointed class or struct is accessible.
• A yellow dashed arrow denotes a relation between a template instance and the template class it was instantiated from. The arrow is labeled with the template parameters of the instance.
# ActionMap Member List

This is the complete list of members for `ActionMap`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Class Name</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>bind</code></td>
<td>ActionMap</td>
</tr>
<tr>
<td><code>bind</code></td>
<td>ActionMap</td>
</tr>
<tr>
<td><code>bindCmd</code></td>
<td>ActionMap</td>
</tr>
<tr>
<td><code>bindObj</code></td>
<td>ActionMap</td>
</tr>
<tr>
<td><code>bindObj</code></td>
<td>ActionMap</td>
</tr>
<tr>
<td><code>call</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clone</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getBinding</code></td>
<td>ActionMap</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCommand(string device, string action)</td>
<td>ActionMap</td>
</tr>
<tr>
<td>getDeadZone(string device, string action)</td>
<td>ActionMap</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getScale(string device, string action)</td>
<td>ActionMap</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInInverted(string device, string action)</td>
<td>ActionMap</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td><code>isMemberOfClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>locked()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>pop()</code></td>
<td>ActionMap</td>
</tr>
<tr>
<td><code>push()</code></td>
<td>ActionMap</td>
</tr>
<tr>
<td><code>save</code></td>
<td>ActionMap</td>
</tr>
<tr>
<td><code>SimObject::save</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>unbind(string device, string action)</td>
<td>ActionMap</td>
</tr>
<tr>
<td>unbindObj(string device, string action, string obj)</td>
<td>ActionMap</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
**AdvancedLightBinManager Member List**

This is the complete list of members for `AdvancedLightBinManager`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code> (SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>binType</code></td>
<td>RenderBinManager</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getBinType()</code></td>
<td>RenderBinManager</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-------------</td>
<td>-------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>processAddOrder</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>renderOrder</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>schedule</td>
<td>float time, string method, string args...</td>
</tr>
<tr>
<td>setCanSave</td>
<td>bool value=true</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>string name</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>bool value=true</td>
</tr>
<tr>
<td>setFieldType</td>
<td>string fieldName, string type</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>string fieldName, string value, int index=-1</td>
</tr>
<tr>
<td>setFilename</td>
<td>string fileName</td>
</tr>
<tr>
<td>setHidden</td>
<td>bool value=true</td>
</tr>
<tr>
<td>setInternalName</td>
<td>string newInternalName</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>bool state=true</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>bool state=true</td>
</tr>
<tr>
<td>setLocked</td>
<td>bool value=true</td>
</tr>
<tr>
<td>setName</td>
<td>string newName</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>bool value=true</td>
</tr>
<tr>
<td>setSuperClassNamespace</td>
<td>string name</td>
</tr>
<tr>
<td>superClass</td>
<td></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# AIclient Member List

This is the complete list of members for `AIclient`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>acceptsAsChild(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>activateGhosting()</code></td>
<td><code>GameConnection</code></td>
</tr>
<tr>
<td><code>add(SimObject objects...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>bringToFront(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>callOnChildren(string method, string args...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>callOnChildrenNoRecurse(string method, string args...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>chaseCam(int size)</code></td>
<td><code>GameConnection</code></td>
</tr>
<tr>
<td><code>checkMaxRate()</code></td>
<td><code>NetConnection</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clear()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>clearCameraObject()</code></td>
<td><code>GameConnection</code></td>
</tr>
<tr>
<td><code>clearPaths()</code></td>
<td><code>NetConnection</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>connect(string remoteAddress)</code></td>
<td><code>NetConnection</code></td>
</tr>
<tr>
<td><code>connectLocal()</code></td>
<td><code>NetConnection</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>delete(string reason=&quot;&quot;)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>NetConnection::delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAddress</td>
<td>AIConnection</td>
</tr>
<tr>
<td>GameConnection::getAddress()</td>
<td>NetConnection</td>
</tr>
<tr>
<td>getAimLocation</td>
<td>AlClient</td>
</tr>
<tr>
<td>getCameraObject()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getControlCameraDefaultFov()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>getControlCameraFov()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>getControlObject()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDamageFlash()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Returns</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>getFieldType</code> (string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue</code> (string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFreeLook</code></td>
<td>AIConnection</td>
</tr>
<tr>
<td><code>getFullCount</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getGhostID</code> (int realID)</td>
<td>NetConnection</td>
</tr>
<tr>
<td><code>getGhostsActive</code></td>
<td>NetConnection</td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getLocation</code></td>
<td>AIClient</td>
</tr>
<tr>
<td><code>getMove</code> (string field)</td>
<td>AIConnection</td>
</tr>
<tr>
<td><code>getMoveDestination</code></td>
<td>AIClient</td>
</tr>
<tr>
<td><code>getName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObject</code> (int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getObjectIndex</code> (SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getPacketLoss</code></td>
<td>NetConnection</td>
</tr>
<tr>
<td><code>getPing</code></td>
<td>NetConnection</td>
</tr>
<tr>
<td><code>getRandom</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getServerConnection</code></td>
<td>GameConnection [static]</td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getTargetObject</code></td>
<td>AIClient</td>
</tr>
<tr>
<td><code>getTrigger</code> (int trigger)</td>
<td>AIConnection</td>
</tr>
<tr>
<td><code>getWhiteOut</code></td>
<td>GameConnection</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>initialControlSet</code></td>
<td>GameConnection</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Function Name</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>isAIControlled()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isControlObjectRotDampedCamera()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>isPlaying()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>isDemoRecording()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstPerson()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>listClassIDs()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>missionCycleCleanup</td>
<td>AIClient</td>
</tr>
<tr>
<td>move</td>
<td>AIClient</td>
</tr>
<tr>
<td>moveForward</td>
<td>AIClient</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onConnectionAccepted()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onConnectionDropped(string reason)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onConnectionError(string errorString)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onConnectionTimedOut()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onConnectRequestRejected(string reason)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>onConnectRequestTimedOut()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onControlObjectChange()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onDataBlocksDone(int sequence)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onDrop(string disconnectReason)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onFlash(bool state)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>play2D(SFXProfile profile)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>play3D(SFXProfile profile, TransformF location)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>playDemo(string demoFileName)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resetGhosting()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>resolveGhostID(int ghostID)</td>
<td>NetConnection</td>
</tr>
<tr>
<td>resolveObjectFromGhostIndex(int ghostID)</td>
<td>NetConnection</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setAimLocation</td>
<td>AIClient</td>
</tr>
<tr>
<td>setBlackOut(bool doFade, int timeMS)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setCameraObject(GameBase camera)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>setClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>setConnectArgs()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setControlCameraFov()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setControlObject()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType()</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue()</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstPerson()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setFreeLook()</td>
<td>AIClient</td>
</tr>
<tr>
<td>setHidden()</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>setJoinPassword()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setLagIcon()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setLocked()</td>
<td>SimObject</td>
</tr>
<tr>
<td>setMissionCRC()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setMove()</td>
<td>AIClient</td>
</tr>
<tr>
<td>setMoveDestination()</td>
<td>AIClient</td>
</tr>
<tr>
<td>setMoveSpeed()</td>
<td>AIClient</td>
</tr>
<tr>
<td>setName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSimulatedNetParams()</td>
<td>NetConnection</td>
</tr>
<tr>
<td>setPacketLoss()</td>
<td>NetConnection</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTargetObject</code></td>
<td><code>AIclient</code></td>
</tr>
<tr>
<td><code>setTrigger(int trigger, bool set)</code></td>
<td><code>AIConnection</code></td>
</tr>
<tr>
<td><code>sort(string callbackFunction)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>startRecording(string fileName)</code></td>
<td><code>GameConnection</code></td>
</tr>
<tr>
<td><code>stop()</code></td>
<td><code>AIclient</code></td>
</tr>
<tr>
<td><code>stopRecording()</code></td>
<td><code>GameConnection</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>transmitDataBlocks(int sequence)</code></td>
<td><code>GameConnection</code></td>
</tr>
<tr>
<td><code>transmitPaths()</code></td>
<td><code>NetConnection</code></td>
</tr>
</tbody>
</table>
### AlConnection Member List

This is the complete list of members for *AlConnection*, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>acceptsAsChild(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>activateGhosting()</code></td>
<td>GameConnection</td>
</tr>
<tr>
<td><code>add(SimObject objects...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>bringToFront(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>callOnChildren(string method, string args...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>callOnChildrenNoRecurse(string method, string args...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>chaseCam(int size)</code></td>
<td>GameConnection</td>
</tr>
<tr>
<td><code>checkMaxRate()</code></td>
<td>NetConnection</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clear()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>clearCameraObject()</code></td>
<td>GameConnection</td>
</tr>
<tr>
<td><code>clearPaths()</code></td>
<td>NetConnection</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>connect(string remoteAddress)</code></td>
<td>NetConnection</td>
</tr>
<tr>
<td><code>connectLocal()</code></td>
<td>NetConnection</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>delete(string reason=&quot;&quot;)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>NetConnection::delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAddress</td>
<td>AIConnection</td>
</tr>
<tr>
<td>GameConnection::getAddress()</td>
<td>NetConnection</td>
</tr>
<tr>
<td>getCameraObject()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getControlCameraDefaultFov()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>getControlCameraFov()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>getControlObject()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDamageFlash()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFreeLook</td>
<td>AIConnection</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGhostID(int realID)</td>
<td>NetConnection</td>
</tr>
<tr>
<td>getGhostsActive()</td>
<td>NetConnection</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMove(string field)</td>
<td>AIConnection</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getPacketLoss()</td>
<td>NetConnection</td>
</tr>
<tr>
<td>getPing()</td>
<td>NetConnection</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getServerConnection()</td>
<td>GameConnection [static]</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTrigger(int trigger)</td>
<td>AIConnection</td>
</tr>
<tr>
<td>getWhiteOut()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>initialControlSet()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isAIControlled()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isControlObjectRotDampedCamera()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>isDemoPlaying()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>isDemoRecording</td>
<td>GameConnection</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstPerson()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>listClassIDs()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onConnectionAccepted()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onConnectionDropped(string reason)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onConnectionError(string errorString)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onConnectionTimedOut()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onConnectRequestRejected(string reason)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onConnectRequestTimedOut()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onControlObjectChange()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onDataBlocksDone(int sequence)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onDrop(string disconnectReason)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onFlash(bool state)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>play2D(SFXProfile profile)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>play3D(SFXProfile profile, TransformF location)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>playDemo(string demoFileName)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resetGhosting()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>resolveGhostID(int ghostID)</td>
<td>NetConnection</td>
</tr>
<tr>
<td>resolveObjectFromGhostIndex(int ghostID)</td>
<td>NetConnection</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setBlackOut(bool doFade, int timeMS)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setCameraObject(GameBase camera)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setConnectArgs(const char *args)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setControlCameraFov(float newFOV)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setControlObject(GameBase ctrlObj)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstPerson(bool firstPerson)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setFreeLook(bool isFreeLook)</td>
<td>AIConnection</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setJoinPassword(string password)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setLagIcon(bool state)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setMissionCRC(int CRC)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setMove(string field, float value)</td>
<td>AIConnection</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSimulatedNetParams(float packetLoss, int delay)</td>
<td>NetConnection</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTrigger(int trigger, bool set)</td>
<td>AIConnection</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>startRecording(string fileName)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>stopRecording()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>transmitDataBlocks(int sequence)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>transmitPaths()</td>
<td>NetConnection</td>
</tr>
</tbody>
</table>
## AIPlayer Member List

This is the complete list of members for **AIPlayer**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>allowAllPoses()</td>
<td>Player</td>
</tr>
<tr>
<td>allowCrouching(bool state)</td>
<td>Player</td>
</tr>
<tr>
<td>allowJetJumping(bool state)</td>
<td>Player</td>
</tr>
<tr>
<td>allowJumping(bool state)</td>
<td>Player</td>
</tr>
<tr>
<td>allowProne(bool state)</td>
<td>Player</td>
</tr>
<tr>
<td>allowSprinting(bool state)</td>
<td>Player</td>
</tr>
<tr>
<td>allowSwimming(bool state)</td>
<td>Player</td>
</tr>
<tr>
<td>applyDamage(float amount)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>applyImpulse(Point3F pos, Point3F vec)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>GameBase::applyImpulse(Point3F pos, VectorF vel)</td>
<td>GameBase</td>
</tr>
<tr>
<td>applyRadialImpulse(Point3F origin, float radius, float magnitude)</td>
<td>GameBase</td>
</tr>
<tr>
<td>applyRepair(float amount)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>boundingBox</td>
<td>GameBase [static]</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canCloak()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>changeMaterial(string mapTo, Material oldMat, Material newMat)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>checkDismountPoint(Point3F oldPos, Point3F pos)</td>
<td>Player</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearAim()</td>
<td>AIPlayer</td>
</tr>
<tr>
<td>clearControlObject()</td>
<td>Player</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>crouchTrigger</td>
<td>Player</td>
</tr>
<tr>
<td>dataBlock</td>
<td>GameBase</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>destroyThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMeshVisibility()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getAimLocation()</td>
<td>AIPlayer</td>
</tr>
<tr>
<td>getAimObject()</td>
<td>AIPlayer</td>
</tr>
<tr>
<td>getAIRepairPoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getControllingClient()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getControllingObject()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getControlObject()</td>
<td>Player</td>
</tr>
<tr>
<td>getDamageFlash()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------</td>
<td>---------------</td>
</tr>
<tr>
<td>getDamageLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageLocation(Point3F pos)</td>
<td>Player</td>
</tr>
<tr>
<td>getDamagePercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageState()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDefaultCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEnergyLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEnergyPercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getEyePoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeTransform()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeVector()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getImageAltTrigger(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageAmmo(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Function Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getLookAtPoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMaxDamage()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getModelFile()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedImage()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedObject()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountSlot()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMoveDestination()</td>
<td>AIPlayer</td>
</tr>
<tr>
<td>getMoveSpeed()</td>
<td>AIPlayer</td>
</tr>
<tr>
<td>getMuzzlePoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzleVector()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getNumDeathAnimations()</td>
<td>Player</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>getPendingImage(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getPose()</code></td>
<td><code>Player</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getRechargeRate()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getRepairRate()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getScale()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getShapeName()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getSkinName()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getSlotTransform(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getState(int slot)</code></td>
<td><code>Player</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTargetCount()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getTargetName(int index)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getVelocity()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getWhiteOut()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hasImageState(int slot, string state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>hidden(int slot, string state)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>imageTrigger0</code></td>
<td><code>Player</code> [static]</td>
</tr>
<tr>
<td><code>imageTrigger1</code></td>
<td><code>Player</code> [static]</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isAIControlled</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>isChildOfGroup(group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isCloaked()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isDestroyed()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isDisabled()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEnabled()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isEqualExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isHidden()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isImageFiring(slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isImageMounted(image)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>AIPlayer</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>AIPlayer</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>jumpJetTrigger</td>
<td>Player</td>
</tr>
<tr>
<td>jumpTrigger</td>
<td>Player</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>maxImpulseVelocity</td>
<td>Player</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>maxPredictionTicks</td>
<td>Player</td>
</tr>
<tr>
<td>maxWarpTicks</td>
<td>Player</td>
</tr>
<tr>
<td>minWarpTicks</td>
<td>Player</td>
</tr>
<tr>
<td>mMoveTolerance</td>
<td>AIPlayer</td>
</tr>
</tbody>
</table>

`mountImage`(ShapeBaseImageData image, int slot, bool loaded=true, string skinTag="")

`mountNode` SceneObject

`mountObject`(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)

`mountPID` SceneObject

`mountPos` SceneObject

`mountRot` SceneObject

`moveStuckTestDelay` AIPlayer

`moveStuckTolerance` AIPlayer

`name` SimObject

`parentGroup` SimObject

`pauseThread`(int slot)

`persistentId` SimObject

`playAudio`(int slot, SFXTrack track)

`playThread`(int slot, string name="")

`position` SceneObject

`proneTrigger` Player [static]

`renderCollision` Player [static]

`renderMyItems` Player [static]

`renderMyPlayer` Player [static]

`rotation` SceneObject

`save`(string fileName, bool
<table>
<thead>
<tr>
<th>Function</th>
<th>Class/Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>selectedOnly=false, string preAppendString=&quot;&quot;  </code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>scale</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>scopeToClient(NetConnection client)</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setActionThread(string name, bool hold=false, bool fsp=true)</code></td>
<td>Player</td>
</tr>
<tr>
<td><code>setAimLocation(Point3F target)</code></td>
<td>AIPlayer</td>
</tr>
<tr>
<td><code>setAimObject(GameBase targetObject, Point3F offset)</code></td>
<td>AIPlayer</td>
</tr>
<tr>
<td><code>setAllMeshesHidden(bool hide)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setArmThread(string name)</code></td>
<td>Player</td>
</tr>
<tr>
<td><code>setCameraFov(float fov)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCloaked(bool cloak)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setControl(bool controlled)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>setControlObject(ShapeBase obj)</code></td>
<td>Player</td>
</tr>
<tr>
<td><code>setDamageFlash(float level)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setDamageLevel(float level)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setDamageState(string state)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setDamageVector(Point3F vec)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setDataBlock(GameBaseData data)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEnergyLevel(float level)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool show)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAltTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAmmo(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageGenericTrigger(int slot, int trigger, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageLoaded(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageScriptAnimPrefix(int slot, string prefix)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTarget(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInvincibleMode(float time, float speed)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setMeshHidden(string name, bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setMoveDestination(Point3F goal, bool slowdown=true)</td>
<td>AIPlayer</td>
</tr>
<tr>
<td>setMoveSpeed(float speed)</td>
<td>AIPlayer</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setRechargeRate(float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setRepairRate(float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setShapeName(string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setSkinName(string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setThreadDir(int slot, bool fwd)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setThreadPosition(int slot, float pos)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setThreadTimeScale(int slot, float scale)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setVelocity(Point3F vel)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setWhiteOut(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>skin</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>sprintTrigger</td>
<td>Player</td>
</tr>
<tr>
<td>startFade(int time, int delay, bool fadeOut)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>stop()</td>
<td>AIPlayer</td>
</tr>
<tr>
<td>stopAudio(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>stopThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>validateCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>vehicleDismountTrigger</td>
<td>Player</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# AITurretShape Member List

This is the complete list of members for AITurretShape, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>activateTurret()</td>
<td>AITurretShape</td>
</tr>
<tr>
<td>addToIgnoreList(ShapeBase obj)</td>
<td>AITurretShape</td>
</tr>
<tr>
<td>applyDamage(float amount)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>applyImpulse(Point3F pos, Point3F vec)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>applyRadialImpulse(Point3F origin, float radius, float magnitude)</td>
<td>GameBase</td>
</tr>
<tr>
<td>applyRepair(float amount)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>boundingBox</td>
<td>GameBase</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canCloak()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>canSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>changeMaterial(string mapTo, Material oldMat, Material newMat)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dataBlock</td>
<td>GameBase</td>
</tr>
<tr>
<td>deactivateTurret()</td>
<td>AITurretShape</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>destroyThread(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>doRespawn()</code></td>
<td><code>TurretShape</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMeshVisibility()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getAIRepairPoint()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getAllowManualFire()</code></td>
<td><code>TurretShape</code></td>
</tr>
<tr>
<td><code>getAllowManualRotation()</code></td>
<td><code>TurretShape</code></td>
</tr>
<tr>
<td><code>getCameraFov()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getControllingClient()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getControllingObject()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamageFlash()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamageLevel()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamagePercent()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamageState()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDataBlock()</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDefaultCameraFov()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEnergyLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEnergyPercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getEyePoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeTransform()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeVector()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getImageAltTrigger(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageAmmo(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageGenericTrigger(int slot, int trigger)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageLoaded(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageScriptAnimPrefix(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageSkinTag(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageState(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageTarget(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageTrigger(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getLastStickyNormal()</code></td>
<td><code>Item</code></td>
</tr>
<tr>
<td><code>getLastStickyPos()</code></td>
<td><code>Item</code></td>
</tr>
<tr>
<td><code>getLookAtPoint(float distance=2000, int typeMask=0xFFFFFFFF)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMaxDamage()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getModelFile()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMountedImage(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountSlot(ShapeBaseImageData image)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMuzzlePoint(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMuzzleVector(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getPendingImage(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getRechargeRate()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getRepairRate()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScale()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getShapeName()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getSkinName()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>getSlotTransform(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getState()</td>
<td>TurretShape</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTarget()</td>
<td>AI_TurretShape</td>
</tr>
<tr>
<td>getTargetCount()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTargetName(int index)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getTurretEulerRotation()</td>
<td>TurretShape</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getVelocity()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getWeaponLeadVelocity()</td>
<td>AI_TurretShape</td>
</tr>
<tr>
<td>getWhiteOut()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hasImageState(int slot, string state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>hasTarget()</td>
<td>AI_TurretShape</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isAIControlled</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isAtRest()</td>
<td>Item</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isCloaked()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isDestroyed()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isDisabled()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEnabled()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isHidden()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isImageFiring(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isImageMounted(ShapeBaseImageData image)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td>AITurretShape [static]</td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isRotating()</code></td>
<td>Item</td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td>AITurretShape [static]</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>isStatic()</code></td>
<td>Item</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>maxWarpTicks</code></td>
<td>Item [static]</td>
</tr>
<tr>
<td><code>minWarpTicks</code></td>
<td>Item [static]</td>
</tr>
<tr>
<td><code>mountImage(ShapeBaseImageData image, int slot, bool loaded=true, string skinTag=&quot;&quot;)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>mountNode</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method/Property</td>
<td>Class/Type</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onEnterLiquid</td>
<td>Item</td>
</tr>
<tr>
<td>onLeaveLiquid</td>
<td>Item</td>
</tr>
<tr>
<td>onStickyCollision</td>
<td>Item</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pauseThread</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAudio</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>playThread</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>recenterTurret</td>
<td>AITurretShape</td>
</tr>
<tr>
<td>removeFromIgnoreList</td>
<td>AITurretShape</td>
</tr>
<tr>
<td>resetTarget</td>
<td>AITurretShape</td>
</tr>
<tr>
<td>respawn</td>
<td>TurretShape</td>
</tr>
<tr>
<td>rotate</td>
<td>Item</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient</td>
<td>NetObject</td>
</tr>
<tr>
<td>setAllGunsFiring</td>
<td>AITurretShape</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>setAllMeshesHidden(bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setAllowManualFire(bool allow)</td>
<td>TurretShape</td>
</tr>
<tr>
<td>setAssignManualRotation(bool allow)</td>
<td>TurretShape</td>
</tr>
<tr>
<td>setCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCloaked(bool cloak)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCollisionTimeout(int ignoreColObj=NULL)</td>
<td>Item</td>
</tr>
<tr>
<td>setControl(bool controlled)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setDamageFlash(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageLevel(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageState(string state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageVector(Point3F vec)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDataBlock(GameBaseData data)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEnergyLevel(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setGunSlotFiring(int slot, bool fire)</td>
<td>AIITurretShape</td>
</tr>
<tr>
<td>setHidden(bool show)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAltTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAmmo(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageGenericTrigger(int slot, int trigger, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageLoaded(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>setImageScriptAnimPrefix(int slot, string prefix)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTarget(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInvincibleMode(float time, float speed)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setMeshHidden(string name, bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setRechargeRate(float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setRepairRate(float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setShapeName(string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setSkinName(string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setThreadDir(int slot, bool fwd)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setThreadPosition(int slot, float pos)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setThreadTimeScale(int slot, float scale)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setTurretEulerRotation(Point3F rot)</td>
<td>TurretShape</td>
</tr>
<tr>
<td>setTurretState(string newState, bool force=false)</td>
<td>AITurretShape</td>
</tr>
<tr>
<td>setVelocity(Point3F vel)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setWeaponLeadVelocity(float velocity)</td>
<td>AITurretShape</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td><code>setWhiteOut(float level)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>skin</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>startFade(int time, int delay, bool fadeOut)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>startScanForTargets()</code></td>
<td><code>AITurretShape</code></td>
</tr>
<tr>
<td><code>startTrackingTarget()</code></td>
<td><code>AITurretShape</code></td>
</tr>
<tr>
<td><code>static</code></td>
<td><code>Item</code></td>
</tr>
<tr>
<td><code>stopAudio(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>stopScanForTargets()</code></td>
<td><code>AITurretShape</code></td>
</tr>
<tr>
<td><code>stopThread(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>stopTrackingTarget()</code></td>
<td><code>AITurretShape</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountImage(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>validateCameraFov(float fov)</code></td>
<td><code>ShapeBase</code></td>
</tr>
</tbody>
</table>
## AITurretShapeData Member List

This is the complete list of members for `AITurretShapeData`, including all inherited members.

<table>
<thead>
<tr>
<th>Method/Property</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code>(<code>SimObject fromObject</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call</code>(<code>string method</code>, <code>string args...</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>cameraDefaultFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMaxDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMaxFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraOffset</code></td>
<td><code>TurretShapeData</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>category</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>checkDeployPos</code>(<code>TransformF txfm</code>)</td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>computeCRC</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cubeReflectorDesc</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>Debris</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>debrisShapeName</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>density</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>destroyedLevel</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td>Member</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>disabledLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>drag</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>elasticity</td>
<td>ItemData</td>
</tr>
<tr>
<td>Explosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>firstPersonOnly</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>friction</td>
<td>ItemData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeployTransform(Point3F pos, Point3F normal)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>gravityMod</td>
<td>ItemData</td>
</tr>
<tr>
<td>headingRate</td>
<td>TurretShapeData</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>inheritEnergyFromMount</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInvincible</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>lightColor</td>
<td>ItemData</td>
</tr>
<tr>
<td>lightOnlyStatic</td>
<td>ItemData</td>
</tr>
<tr>
<td>lightRadius</td>
<td>ItemData</td>
</tr>
<tr>
<td>lightTime</td>
<td>ItemData</td>
</tr>
<tr>
<td>lightType</td>
<td>ItemData</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mass</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>maxDamage</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>maxEnergy</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>maxHeading</td>
<td>TurretShapeData</td>
</tr>
<tr>
<td>maxPitch</td>
<td>TurretShapeData</td>
</tr>
<tr>
<td>maxScanDistance</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>Function</td>
<td>Module/Class</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>maxScanHeading</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>maxScanPitch</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>maxVelocity</td>
<td>ItemData</td>
</tr>
<tr>
<td>maxWeaponRange</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>minPitch</td>
<td>TurretShapeData</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>observeThroughObject</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onAdd(obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onCollision(obj, collObj, vec, len)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onDamage(obj, delta)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onDestroyed(obj, lastState)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onDisabled(obj, lastState)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onEnabled(obj, lastState)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onEndSequence(obj, slot)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onForceUncloak(obj, reason)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onImpact(obj, collObj, vec, len)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onMount(obj, mountObj, node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onMountObject(turret, obj, node)</td>
<td>TurretShapeData</td>
</tr>
<tr>
<td>onNewDataBlock(obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onRemove(obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onStickyCollision(obj)</td>
<td>TurretShapeData</td>
</tr>
<tr>
<td>onTrigger(obj, index, state)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onUnmount(obj, obj)</td>
<td>GameBaseData</td>
</tr>
</tbody>
</table>
mountObj, int node

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>onUnmountObject</strong></td>
<td>(TurretShape turret, SceneObject obj)</td>
</tr>
<tr>
<td><strong>parentGroup</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>persistentId</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>pitchRate</strong></td>
<td>TurretShapeData</td>
</tr>
<tr>
<td><strong>reloadOnLocalClient</strong></td>
<td>()</td>
</tr>
<tr>
<td><strong>renderWhenDestroyed</strong></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><strong>repairRate</strong></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><strong>save</strong></td>
<td>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
</tr>
<tr>
<td><strong>scanTickFrequency</strong></td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td><strong>scanTickFrequencyVariance</strong></td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td><strong>schedule</strong></td>
<td>(float time, string method, string args...)</td>
</tr>
<tr>
<td><strong>setCanSave</strong></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><strong>setClassNamespace</strong></td>
<td>(string name)</td>
</tr>
<tr>
<td><strong>setEditorOnly</strong></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><strong>setFieldType</strong></td>
<td>(string fieldName, string type)</td>
</tr>
<tr>
<td><strong>setFieldValue</strong></td>
<td>(string fieldName, string value, int index=-1)</td>
</tr>
<tr>
<td><strong>setFilename</strong></td>
<td>(string fileName)</td>
</tr>
<tr>
<td><strong>setHidden</strong></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><strong>setInternalName</strong></td>
<td>(string newInternalName)</td>
</tr>
<tr>
<td><strong>setIsExpanded</strong></td>
<td>(bool state=true)</td>
</tr>
<tr>
<td><strong>setIsSelected</strong></td>
<td>(bool state=true)</td>
</tr>
<tr>
<td><strong>setLocked</strong></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><strong>setName</strong></td>
<td>(string newName)</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>shadowEnable</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowMaxVisibleDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowProjectionDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSize</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSphereAdjust</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shapeFile</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>simpleServerCollision</td>
<td>ItemData</td>
</tr>
<tr>
<td>startLoaded</td>
<td>TurretShapeData</td>
</tr>
<tr>
<td>stateDirection</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>stateFire</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>stateName</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>stateScaleAnimation</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>stateScan</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>stateScript</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>stateSequence</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>stateTimeoutValue</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>stateTransitionOnActivated</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>stateTransitionOnAtRest</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>stateTransitionOnDeactivated</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>stateTransitionOnNoTarget</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>stateTransitionOnNotAtRest</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>stateTransitionOnTarget</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>stateTransitionOnTimeout</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>stateWaitForTimeout</td>
<td>AITurretShapeData</td>
</tr>
<tr>
<td>sticky</td>
<td>ItemData</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>trackLostTargetTime</td>
<td>AI TurretShapeData</td>
</tr>
<tr>
<td>underwaterExplosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>useEyePoint</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>weaponLeadVelocity</td>
<td>AI TurretShapeData</td>
</tr>
<tr>
<td>weaponLinkType</td>
<td>TurretShapeData</td>
</tr>
<tr>
<td>zRotOnly</td>
<td>TurretShapeData</td>
</tr>
</tbody>
</table>
ArrayObject Member List

This is the complete list of members for ArrayObject, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>add</strong>(string key, string value=&quot;&quot;)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><strong>append</strong>(ArrayObject target)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><strong>assignFieldsFrom</strong>(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>assignPersistentId</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>call</strong>(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>canSave</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>canSaveDynamicFields</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>caseSensitive</strong></td>
<td>ArrayObject</td>
</tr>
<tr>
<td><strong>class</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>className</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>clone</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>count</strong></td>
<td>ArrayObject</td>
</tr>
<tr>
<td><strong>countKey</strong>(string key)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><strong>countValue</strong>(string value)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><strong>crop</strong>(ArrayObject target)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><strong>deepClone</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>delete</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>dump</strong>(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>dumpClassHierarchy</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>dumpGroupHierarchy</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>dumpMethods</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>duplicate</strong>(ArrayObject target)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><strong>echo</strong></td>
<td>ArrayObject</td>
</tr>
<tr>
<td><strong>empty</strong></td>
<td>ArrayObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>erase(int index)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCurrent()</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getIndexFromKey(string key)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>getIndexFromValue(string value)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getKey(int index)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(int index)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>insert(string key, string value, int index)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>key</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>moveFirst()</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>moveLast()</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>moveNext()</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>movePrev()</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pop_back()</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>pop_front()</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>push_back(string key, string value=&quot;&quot;&quot;&quot;)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>push_front(string key, string value=&quot;&quot;&quot;&quot;)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCurrent(int index)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setKey</code> (string key, int index)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setValue</code> (string value, int index)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><code>sort</code> (bool ascending=false)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><code>sorta</code> ()</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><code>sortd</code> ()</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><code>sortf</code> (string functionName)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><code>sortfd</code> (string functionName)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><code>sortfk</code> (string functionName)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><code>sortfkd</code> (string functionName)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><code>sortk</code> (bool ascending=false)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><code>sortka</code> ()</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><code>sortkd</code> ()</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><code>sortn</code> (bool ascending=false)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><code>sortna</code> ()</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><code>sortnd</code> ()</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><code>sortnk</code> (bool ascending=false)</td>
<td>ArrayObject</td>
</tr>
<tr>
<td><code>sortnka</code> ()</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>sortnkd()</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>uniqueKey()</td>
<td>ArrayObject</td>
</tr>
<tr>
<td>uniqueValue()</td>
<td>ArrayObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## BanList Member List

This is the complete list of members for BanList, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>add</strong> (int uniqueld, string transportAddress, int banLength)</td>
<td>Add member to ban list with unique ID and transport address</td>
<td>BanList [static]</td>
</tr>
<tr>
<td><strong>addAbsolute</strong> (int uniqueld, string transportAddress, int banTime)</td>
<td>Add absolute ban to member with unique ID and transport address</td>
<td>BanList [static]</td>
</tr>
<tr>
<td><strong>export</strong> (string filename)</td>
<td>Export ban list to file</td>
<td>BanList [static]</td>
</tr>
<tr>
<td><strong>isBanned</strong> (int uniqueld, string transportAddress)</td>
<td>Check if member is banned</td>
<td>BanList [static]</td>
</tr>
<tr>
<td><strong>removeBan</strong> (int uniqueld, string transportAddress)</td>
<td>Remove ban from member with unique ID and transport address</td>
<td>BanList [static]</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
BasicClouds Member List

This is the complete list of members for BasicClouds, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getScale()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method/Property</td>
<td>Class/Type</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>height</td>
<td>BasicClouds</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>BasicClouds [static]</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>BasicClouds [static]</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>layerEnabled</td>
<td>BasicClouds</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>mountRot</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>position</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>rotation</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>save</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>scopeToClient</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setScale</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setScopeAlways</code></td>
<td>NetObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>texDirection</td>
<td>BasicClouds</td>
</tr>
<tr>
<td>texOffset</td>
<td>BasicClouds</td>
</tr>
<tr>
<td>texScale</td>
<td>BasicClouds</td>
</tr>
<tr>
<td>texSpeed</td>
<td>BasicClouds</td>
</tr>
<tr>
<td>texture</td>
<td>BasicClouds</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# Camera Member List

This is the complete list of members for `Camera`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>angularDrag</code></td>
<td>Camera</td>
</tr>
<tr>
<td><code>angularForce</code></td>
<td>Camera</td>
</tr>
<tr>
<td><code>applyDamage(float amount)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>applyImpulse(Point3F pos, Point3F vec)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>GameBase::applyImpulse(Point3F pos, VectorF vel)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>applyRadialImpulse(Point3F origin, float radius, float magnitude)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>applyRepair(float amount)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>autoFitRadius(float radius)</code></td>
<td>Camera</td>
</tr>
<tr>
<td><code>boundingBox</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>brakeMultiplier</code></td>
<td>Camera</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canCloak()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>changeMaterial(string mapTo, Material oldMat, Material newMat)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>controlMode</code></td>
<td>Camera</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>dataBlock</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>destroyThread(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>drag</code></td>
<td><code>Camera</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMeshVisibility()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>force</code></td>
<td><code>Camera</code></td>
</tr>
<tr>
<td><code>getAIRepairPoint()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getAngularVelocity()</code></td>
<td><code>Camera</code></td>
</tr>
<tr>
<td><code>getCameraFov()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getControllingClient()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getControllingObject()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamageFlash()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamageLevel()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamagePercent()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamageState()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDataBlock()</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDefaultCameraFov()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getEnergyLevel()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getEnergyPercent()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getEulerRotation()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getEyePoint()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getEyeTransform()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getEyeVector()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getImageAltTrigger(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageAmmo(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageGenericTrigger(int slot, int trigger)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageLoaded(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageScriptAnimPrefix(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageSkinTag(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageState(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageTarget(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageTrigger(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getLookAtPoint(float distance=2000, int typeMask=0xFFFFFFFF)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMaxDamage()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMode()</td>
<td>Camera</td>
</tr>
<tr>
<td>getModelFile()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountSlot(ShapeBaseImageData image)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzlePoint(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzleVector(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getOffset()</td>
<td>Camera</td>
</tr>
<tr>
<td>getPendingImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getPosition(int slot)</td>
<td>Camera</td>
</tr>
<tr>
<td>getRechargeRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRepairRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRotation()</td>
<td>Camera</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getShapeName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>getSkinName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSlotTransform(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTargetCount()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTargetName(int index)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getVelocity()</td>
<td>Camera</td>
</tr>
<tr>
<td>getWhiteOut()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hasImageState(int slot, string state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isAIControlled</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isCloaked()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isDestroyed()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isDisabled()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isEditOrbitMode()</td>
<td>Camera</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEnabled()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isHidden()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>isImageFiring</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageMounted</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>Camera</code></td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isRotationDamped</code></td>
<td><code>Camera</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>Camera</code></td>
</tr>
<tr>
<td><code>isSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>lookAt(Point3F point)</code></td>
<td><code>Camera</code></td>
</tr>
<tr>
<td><code>mass</code></td>
<td><code>Camera</code></td>
</tr>
<tr>
<td><code>mountImage</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>mountNode</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountObject</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountPID</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountPos</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountRot</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>movementSpeed</code></td>
<td><code>Camera</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>newtonMode</td>
<td>Camera</td>
</tr>
<tr>
<td>newtonRotation</td>
<td>Camera</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pauseThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>persistentld</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAudio(int slot, SFXTrack track)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>playThread(int slot, string name=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setAllMeshesHidden(bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setAngularDrag(float drag)</td>
<td>Camera</td>
</tr>
<tr>
<td>setAngularForce(float force)</td>
<td>Camera</td>
</tr>
<tr>
<td>setAngularVelocity(VectorF velocity)</td>
<td>Camera</td>
</tr>
<tr>
<td>setBrakeMultiplier(float multiplier)</td>
<td>Camera</td>
</tr>
<tr>
<td>setCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCloaked(bool cloak)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setControl(bool controlled)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setDamageFlash(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageLevel(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageState(string state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td><code>setDamageVector(Point3F vec)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setDataBlock(GameBaseData data)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>setDrag(float drag)</code></td>
<td>Camera</td>
</tr>
<tr>
<td><code>setEditOrbitMode()</code></td>
<td>Camera</td>
</tr>
<tr>
<td><code>setEditOrbitPoint(Point3F point)</code></td>
<td>Camera</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEnergyLevel(float level)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFlyForce(float force)</code></td>
<td>Camera</td>
</tr>
<tr>
<td><code>setFlyMode()</code></td>
<td>Camera</td>
</tr>
<tr>
<td><code> setHidden(bool show)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setImageAltTrigger(int slot, bool state)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setImageAmmo(int slot, bool state)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setImageGenericTrigger(int slot, int trigger, bool state)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setImageLoaded(int slot, bool state)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setImageScriptAnimPrefix(int slot, string prefix)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setImageTarget(int slot, bool state)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setImageTrigger(int slot, bool state)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInvincibleMode(float time, float speed)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setMass(float mass)</code></td>
<td>Camera</td>
</tr>
<tr>
<td>Function Name</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>setMeshHidden</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNewtonFlyMode</code></td>
<td><code>Camera</code></td>
</tr>
<tr>
<td><code>setOffset</code></td>
<td><code>Camera</code></td>
</tr>
<tr>
<td><code>setOrbitMode</code></td>
<td><code>Camera</code></td>
</tr>
<tr>
<td><code>setOrbitObject</code></td>
<td><code>Camera</code></td>
</tr>
<tr>
<td><code>setOrbitPoint</code></td>
<td><code>Camera</code></td>
</tr>
<tr>
<td><code>setRechargeRate</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setRepairRate</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setRotation</code></td>
<td><code>Camera</code></td>
</tr>
<tr>
<td><code>setScale</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setShapeName</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setSkinName</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setSpeedMultiplier</code></td>
<td><code>Camera</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setThreadDir</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>setThreadPosition(int slot, float pos)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setThreadTimeScale(int slot, float scale)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setTrackObject(GameBase trackObject, Point3F offset=Point3F(0.0f, 0.0f, 0.0f))</td>
<td>Camera</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setValidEditOrbitPoint(bool validPoint)</td>
<td>Camera</td>
</tr>
<tr>
<td>setVelocity(VectorF velocity)</td>
<td>Camera</td>
</tr>
<tr>
<td>ShapeBase::setVelocity(Point3F vel)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setWhiteOut(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>skin</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>speedMultiplier</td>
<td>Camera</td>
</tr>
<tr>
<td>startFade(int time, int delay, bool fadeOut)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>stopAudio(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>stopThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>validateCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
</tbody>
</table>
# CameraData Member List

This is the complete list of members for `CameraData`, including all inherited members.

<table>
<thead>
<tr>
<th>Method/Selector</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>cameraDefaultFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMaxDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMaxFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>category</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>checkDeployPos(TransformF txfm)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>computeCRC</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cubeReflectorDesc</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>Debris</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>debrisShapeName</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>density</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>destroyedLevel</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>disabledLevel</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>drag</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Explosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>firstPersonOnly</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeployTransform(Point3F pos, Point3F normal)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>inheritEnergyFromMount</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>Method/Property</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInvincible</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>mass</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>maxDamage</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>maxEnergy</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>observeThroughObject</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onAdd(GameBase obj)</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onCollision(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onDamage(ShapeBase obj, float delta)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onDestroyed(ShapeBase obj, string lastState)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onDisabled(ShapeBase obj, string lastState)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onEnabled(ShapeBase obj, string lastState)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onEndSequence(ShapeBase obj, int slot)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onForceUncloak(ShapeBase obj, string reason)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onImpact(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td><strong>onMount</strong></td>
<td>(GameBase obj, SceneObject mountObj, int node)</td>
</tr>
<tr>
<td><strong>onNewDataBlock</strong></td>
<td>(GameBase obj)</td>
</tr>
<tr>
<td><strong>onRemove</strong></td>
<td>(GameBase obj)</td>
</tr>
<tr>
<td><strong>onTrigger</strong></td>
<td>(ShapeBase obj, int index, bool state)</td>
</tr>
<tr>
<td><strong>onUnmount</strong></td>
<td>(GameBase obj, SceneObject mountObj, int node)</td>
</tr>
<tr>
<td><strong>parentGroup</strong></td>
<td></td>
</tr>
<tr>
<td><strong>persistentId</strong></td>
<td></td>
</tr>
<tr>
<td><strong>reloadOnLocalClient</strong></td>
<td>()</td>
</tr>
<tr>
<td><strong>renderWhenDestroyed</strong></td>
<td></td>
</tr>
<tr>
<td><strong>repairRate</strong></td>
<td></td>
</tr>
<tr>
<td><strong>save</strong></td>
<td>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
</tr>
<tr>
<td><strong>schedule</strong></td>
<td>(float time, string method, string args...)</td>
</tr>
<tr>
<td><strong>setCanSave</strong></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><strong>setClassNameNamespace</strong></td>
<td>(string name)</td>
</tr>
<tr>
<td><strong>setEditorOnly</strong></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><strong>setFieldType</strong></td>
<td>(string fieldName, string type)</td>
</tr>
<tr>
<td><strong>setFieldValue</strong></td>
<td>(string fieldName, string value, int index=-1)</td>
</tr>
<tr>
<td><strong>setFilename</strong></td>
<td>(string fileName)</td>
</tr>
<tr>
<td><strong>setHidden</strong></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><strong>setInternalName</strong></td>
<td>(string newInternalName)</td>
</tr>
<tr>
<td><strong>setIsExpanded</strong></td>
<td>(bool state=true)</td>
</tr>
<tr>
<td><strong>setIsSelected</strong></td>
<td>(bool state=true)</td>
</tr>
<tr>
<td><strong>setLocked</strong></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><strong>setName</strong></td>
<td>(string newName)</td>
</tr>
<tr>
<td><strong>setNameChangeAllowed</strong></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>shadowEnable</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>shadowMaxVisibleDistance</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>shadowProjectionDistance</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>shadowSize</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>shadowSphereAdjust</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>shapeFile</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>underwaterExplosion</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>useEyePoint</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
</tbody>
</table>
CloudLayer Member List

This is the complete list of members for CloudLayer, including all inherited members.

<table>
<thead>
<tr>
<th>Method/Property</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>baseColor</td>
<td>CloudLayer</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>coverage</td>
<td>CloudLayer</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>exposure</td>
<td>CloudLayer</td>
</tr>
<tr>
<td>getCansave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class/Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>height</td>
<td>CloudLayer</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>CloudLayer [static]</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>CloudLayer [static]</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient (NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType (string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>texDirection</code></td>
<td><code>CloudLayer</code></td>
</tr>
<tr>
<td><code>texScale</code></td>
<td><code>CloudLayer</code></td>
</tr>
<tr>
<td><code>texSpeed</code></td>
<td><code>CloudLayer</code></td>
</tr>
<tr>
<td><code>texture</code></td>
<td><code>CloudLayer</code></td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>windSpeed</code></td>
<td><code>CloudLayer</code></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
### ConsoleLogger Member List

This is the complete list of members for ConsoleLogger, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>attach()</td>
<td>ConsoleLogger</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>detach()</td>
<td>ConsoleLogger</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>level</td>
<td>ConsoleLogger</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString='&quot;&quot;')</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
**ConvexShape Member List**

This is the complete list of members for *ConvexShape*, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>ConvexShape</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>ConvexShape</code></td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>Material</code></td>
<td><code>ConvexShape</code></td>
</tr>
<tr>
<td><code>mountNode</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountPID</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountPos</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td><code>mountRot</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>position</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>rotation</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString='')</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>scale</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>scopeToClient(NetConnection client)</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>surface</td>
<td>ConvexShape</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
</tbody>
</table>
This is the complete list of members for **CubemapData**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code></td>
<td>SimObject fromObject</td>
</tr>
<tr>
<td><code>assignPersistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>call</code></td>
<td>String method, string args...</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clone</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>cubeFace</code></td>
<td>CubemapData</td>
</tr>
<tr>
<td><code>deepClone</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump</code></td>
<td>Boolean detailed=false</td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dynamic</code></td>
<td>CubemapData</td>
</tr>
<tr>
<td><code>dynamicFarDist</code></td>
<td>CubemapData</td>
</tr>
<tr>
<td><code>dynamicNearDist</code></td>
<td>CubemapData</td>
</tr>
<tr>
<td><code>dynamicObjectTypeMask</code></td>
<td>CubemapData</td>
</tr>
<tr>
<td><code>dynamicSize</code></td>
<td>CubemapData</td>
</tr>
<tr>
<td><code>getCanSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>CubemapData</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getld()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentld</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>updateFaces()</code></td>
<td><code>CubemapData</code></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
CustomMaterial Member List

This is the complete list of members for CustomMaterial, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>alphaRef</td>
<td>Material</td>
</tr>
<tr>
<td>alphaTest</td>
<td>Material</td>
</tr>
<tr>
<td>animFlags</td>
<td>Material</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>baseTex</td>
<td>Material</td>
</tr>
<tr>
<td>bumpAtlas</td>
<td>Material</td>
</tr>
<tr>
<td>bumpTex</td>
<td>Material</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>castShadows</td>
<td>Material</td>
</tr>
<tr>
<td>cellIndex</td>
<td>Material</td>
</tr>
<tr>
<td>cellLayout</td>
<td>Material</td>
</tr>
<tr>
<td>cellSize</td>
<td>Material</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>colorMultiply</td>
<td>Material</td>
</tr>
<tr>
<td>cubemap</td>
<td>Material</td>
</tr>
<tr>
<td>customFootstepSound</td>
<td>Material</td>
</tr>
<tr>
<td>customImpactSound</td>
<td>Material</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>detailMap</td>
<td>Material</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>detailNormalMap</td>
<td>Material</td>
</tr>
<tr>
<td>detailNormalMapStrength</td>
<td>Material</td>
</tr>
<tr>
<td>detailScale</td>
<td>Material</td>
</tr>
<tr>
<td>detailTex</td>
<td>Material</td>
</tr>
<tr>
<td>diffuseColor</td>
<td>Material</td>
</tr>
<tr>
<td>diffuseMap</td>
<td>Material</td>
</tr>
<tr>
<td>doubleSided</td>
<td>Material</td>
</tr>
<tr>
<td><strong>dump</strong> (bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpInstances</td>
<td>Material</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dynamicCubemap</td>
<td>Material</td>
</tr>
<tr>
<td>effectColor</td>
<td>Material</td>
</tr>
<tr>
<td>emissive</td>
<td>Material</td>
</tr>
<tr>
<td>envMap</td>
<td>Material</td>
</tr>
<tr>
<td>envTex</td>
<td>Material</td>
</tr>
<tr>
<td>fallback</td>
<td>CustomMaterial</td>
</tr>
<tr>
<td>flush</td>
<td>Material</td>
</tr>
<tr>
<td>footstepSoundId</td>
<td>Material</td>
</tr>
<tr>
<td>forwardLit</td>
<td>CustomMaterial</td>
</tr>
<tr>
<td>getAnimFlags</td>
<td>Material</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td>Material</td>
</tr>
<tr>
<td><code>SimObject::getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>glow</code></td>
<td>Material</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>impactSoundId</code></td>
<td>Material</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>Material</td>
</tr>
<tr>
<td><code>isAutoGenerated</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>lightMap</code></td>
<td>Material</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>mapTo</td>
<td>Material</td>
</tr>
<tr>
<td>minnaertConstant</td>
<td>Material</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>normalMap</td>
<td>Material</td>
</tr>
<tr>
<td>overlayMap</td>
<td>Material</td>
</tr>
<tr>
<td>overlayTex</td>
<td>Material</td>
</tr>
<tr>
<td>parallaxScale</td>
<td>Material</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pixelSpecular</td>
<td>Material</td>
</tr>
<tr>
<td>planarReflection</td>
<td>Material</td>
</tr>
<tr>
<td>reload</td>
<td>Material</td>
</tr>
<tr>
<td>rotPivotOffset</td>
<td>Material</td>
</tr>
<tr>
<td>rotSpeed</td>
<td>Material</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>scrollDir</td>
<td>Material</td>
</tr>
<tr>
<td>scrollSpeed</td>
<td>Material</td>
</tr>
<tr>
<td>sequenceFramePerSec</td>
<td>Material</td>
</tr>
<tr>
<td>sequenceSegmentSize</td>
<td>Material</td>
</tr>
<tr>
<td>setAutoGenerated</td>
<td>Material</td>
</tr>
<tr>
<td>setCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method/Property</td>
<td>Type</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td><code>index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>shader</code></td>
<td><code>CustomMaterial</code></td>
</tr>
<tr>
<td><code>showDust</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td><code>showFootprints</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td><code>specular</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td><code>specularMap</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td><code>specularPower</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td><code>stateBlock</code></td>
<td><code>CustomMaterial</code></td>
</tr>
<tr>
<td><code>subSurface</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td><code>subSurfaceColor</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td><code>subSurfaceRolloff</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>target</code></td>
<td><code>CustomMaterial</code></td>
</tr>
<tr>
<td><code>toneMap</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td><code>translucent</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td><code>translucentBlendOp</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td><code>translucentZWrite</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td><code>useAnisotropic</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td>version</td>
<td>CustomMaterial</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------</td>
</tr>
<tr>
<td>vertColor</td>
<td>Material</td>
</tr>
<tr>
<td>vertLit</td>
<td>Material</td>
</tr>
<tr>
<td>waveAmp</td>
<td>Material</td>
</tr>
<tr>
<td>waveFreq</td>
<td>Material</td>
</tr>
<tr>
<td>waveType</td>
<td>Material</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
Debris Member List

This is the complete list of members for Debris, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>applyImpulse(Point3F pos, VectorF vel)</td>
<td>GameBase</td>
</tr>
<tr>
<td>applyRadialImpulse(Point3F origin, float radius, float magnitude)</td>
<td>GameBase</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>boundingBox</td>
<td>GameBase</td>
</tr>
<tr>
<td>[static] call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dataBlock</td>
<td>GameBase</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>init</code></td>
<td><code>Debris</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>Debris</code></td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>Debris</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td>lifetime</td>
<td>Debris</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCansave</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setControl</td>
<td>GameBase</td>
</tr>
<tr>
<td>setDatavblock</td>
<td>GameBase</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldtype</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>setHidden</td>
<td>Set hidden state</td>
</tr>
<tr>
<td>setInternalName</td>
<td>Set internal name</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>Set is expanded state</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>Set is selected state</td>
</tr>
<tr>
<td>setLocked</td>
<td>Set locked state</td>
</tr>
<tr>
<td>setName</td>
<td>Set name</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>Set name change allowed state</td>
</tr>
<tr>
<td>setScale</td>
<td>Set scale</td>
</tr>
<tr>
<td>setScopeAlways</td>
<td>Set scope always</td>
</tr>
<tr>
<td>setSuperClassNameSpace</td>
<td>Set superclass namespace</td>
</tr>
<tr>
<td>setTransform</td>
<td>Set transform</td>
</tr>
<tr>
<td>superClass</td>
<td>Get superclass</td>
</tr>
<tr>
<td>unmount</td>
<td>Unmount object</td>
</tr>
<tr>
<td>unmountObject</td>
<td>Unmount object</td>
</tr>
</tbody>
</table>
### DebrisData Member List

This is the complete list of members for `DebrisData`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>baseRadius</td>
<td>DebrisData</td>
</tr>
<tr>
<td>bounceVariance</td>
<td>DebrisData</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>category</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>elasticity</td>
<td>DebrisData</td>
</tr>
<tr>
<td>emitters</td>
<td>DebrisData</td>
</tr>
<tr>
<td>explodeOnMaxBounce</td>
<td>DebrisData</td>
</tr>
<tr>
<td>Explosion</td>
<td>DebrisData</td>
</tr>
<tr>
<td>fade</td>
<td>DebrisData</td>
</tr>
<tr>
<td>friction</td>
<td>DebrisData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class Type</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>gravModifier</code></td>
<td>DebrisData</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>ignoreWater</code></td>
<td>DebrisData</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Module</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>lifetime</td>
<td>DebrisData</td>
</tr>
<tr>
<td>lifetimeVariance</td>
<td>DebrisData</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>maxSpinSpeed</td>
<td>DebrisData</td>
</tr>
<tr>
<td>minSpinSpeed</td>
<td>DebrisData</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>numBounces</td>
<td>DebrisData</td>
</tr>
<tr>
<td>onAdd(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onMount(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onNewDataBlock(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onRemove(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onUnmount(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>shapeFile</code></td>
<td><code>DebrisData</code></td>
</tr>
<tr>
<td><code>snapOnMaxBounce</code></td>
<td><code>DebrisData</code></td>
</tr>
<tr>
<td><code>staticOnMaxBounce</code></td>
<td><code>DebrisData</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>terminalVelocity</code></td>
<td><code>DebrisData</code></td>
</tr>
<tr>
<td><code>texture</code></td>
<td><code>DebrisData</code></td>
</tr>
<tr>
<td><code>useRadiusMass</code></td>
<td><code>DebrisData</code></td>
</tr>
<tr>
<td><code>velocity</code></td>
<td><code>DebrisData</code></td>
</tr>
<tr>
<td><code>velocityVariance</code></td>
<td><code>DebrisData</code></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# DebugDrawer Member List

This is the complete list of members for `DebugDrawer`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>drawBox(Point3F a, Point3F b, ColorF color=ColorF::WHITE)</code></td>
<td><code>DebugDrawer</code></td>
</tr>
<tr>
<td><code>drawLine(Point3F a, Point3F b, ColorF color=ColorF::WHITE)</code></td>
<td><code>DebugDrawer</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLastTTL(int ms)</td>
<td>DebugDrawer</td>
</tr>
<tr>
<td>setLastZTest(bool enabled)</td>
<td>DebugDrawer</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>toggleDrawing()</td>
<td>DebugDrawer</td>
</tr>
<tr>
<td>toggleFreeze()</td>
<td>DebugDrawer</td>
</tr>
</tbody>
</table>
# DecalData Member List

This is the complete list of members for DecalData, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clippingAngle</code></td>
<td>DecalData</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>fadeEndPixelSize</code></td>
<td>DecalData</td>
</tr>
<tr>
<td><code>fadeStartPixelSize</code></td>
<td>DecalData</td>
</tr>
<tr>
<td><code>fadeTime</code></td>
<td>DecalData</td>
</tr>
<tr>
<td><code>frame</code></td>
<td>DecalData</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>lifeSpan</td>
<td>DecalData</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>Material</td>
<td>DecalData</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class/Type</td>
</tr>
<tr>
<td>------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>postApply()</td>
<td>DecalData</td>
</tr>
<tr>
<td>randomize</td>
<td>DecalData</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>renderPriority</td>
<td>DecalData</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>size</td>
<td>DecalData</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>texCols</td>
<td>DecalData</td>
</tr>
<tr>
<td>texRows</td>
<td>DecalData</td>
</tr>
<tr>
<td>textureCoordCount</td>
<td>DecalData</td>
</tr>
<tr>
<td>textureCoords</td>
<td>DecalData</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# DecalManager Member List

This is the complete list of members for **DecalManager**, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>getEulerRotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable DecalManager [static]</td>
<td></td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable DecalManager [static]</td>
<td></td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td><code>setSuperClassNamespace()</code></td>
<td>(string name)</td>
</tr>
<tr>
<td><code>setTransform()</code></td>
<td>(TransformF txfm)</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td></td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td></td>
</tr>
<tr>
<td><code>unmountObject()</code></td>
<td>(SceneObject target)</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
**DecalRoad Member List**

This is the complete list of members for **DecalRoad**, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>breakAngle</td>
<td>DecalRoad</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>discardAll</td>
<td>DecalRoad</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>EditorOpen</td>
<td>DecalRoad</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>getDynamicField</code>(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getEulerRotation</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getField</code>(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType</code>(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue</code>(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getForwardVector</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getGhostID</code>()</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getGroup</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInverseTransform</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountedObject</code>(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountedObjectCount</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountedObjectNode</code>(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountNodeObject</code>(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getName</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObjectBox</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getObjectMount</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getPosition</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getRightVector</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getScale</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getServerObject</code>()</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getTransform</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function/Method</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>DecalRoad</code></td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>DecalRoad</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>Material</code></td>
<td><code>DecalRoad</code></td>
</tr>
<tr>
<td><code>mountNode</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>Node</td>
<td>DecalRoad</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>postApply()</td>
<td>DecalRoad</td>
</tr>
<tr>
<td>regenerate()</td>
<td>DecalRoad</td>
</tr>
<tr>
<td>renderPriority</td>
<td>DecalRoad</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setScale</code> (Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setScopeAlways</code> ()</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setTransform</code> (TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>showBatches</code></td>
<td>DecalRoad</td>
</tr>
<tr>
<td></td>
<td>[static]</td>
</tr>
<tr>
<td><code>showRoad</code></td>
<td>DecalRoad</td>
</tr>
<tr>
<td></td>
<td>[static]</td>
</tr>
<tr>
<td><code>showSpline</code></td>
<td>DecalRoad</td>
</tr>
<tr>
<td></td>
<td>[static]</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>textureLength</code></td>
<td>DecalRoad</td>
</tr>
<tr>
<td><code>unmount</code> ()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>unmountObject</code> (SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>updateDelay</code></td>
<td>DecalRoad</td>
</tr>
<tr>
<td></td>
<td>[static]</td>
</tr>
<tr>
<td><code>wireframe</code></td>
<td>DecalRoad</td>
</tr>
<tr>
<td></td>
<td>[static]</td>
</tr>
</tbody>
</table>
**EventManager Member List**

This is the complete list of members for *EventManager*, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpEvents()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpSubscribers(String event)</td>
<td>EventManager</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>getFieldCount</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(fieldName, index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRegisteredEvent(event)</td>
<td>EventManager</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>postEvent(event, data)</td>
<td>EventManager</td>
</tr>
<tr>
<td>queue</td>
<td>EventManager</td>
</tr>
<tr>
<td>registerEvent(event)</td>
<td>EventManager</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>remove()</code></td>
<td>(SimObject listener, String event) EventManager</td>
</tr>
<tr>
<td><code>removeAll()</code></td>
<td>(SimObject listener) EventManager</td>
</tr>
<tr>
<td><code>save()</code></td>
<td>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;) SimObject</td>
</tr>
<tr>
<td><code>schedule()</code></td>
<td>(float time, string method, string args...) SimObject</td>
</tr>
<tr>
<td><code>setCanSave()</code></td>
<td>(bool value=true) SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace()</code></td>
<td>(string name) SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly()</code></td>
<td>(bool value=true) SimObject</td>
</tr>
<tr>
<td><code>setFieldType()</code></td>
<td>(string fieldName, string type) SimObject</td>
</tr>
<tr>
<td><code>setFieldValue()</code></td>
<td>(string fieldName, string value, int index=-1) SimObject</td>
</tr>
<tr>
<td><code>setFilename()</code></td>
<td>(string fileName) SimObject</td>
</tr>
<tr>
<td><code>setHidden()</code></td>
<td>(bool value=true) SimObject</td>
</tr>
<tr>
<td><code>setInternalName()</code></td>
<td>(string newInternalName) SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded()</code></td>
<td>(bool state=true) SimObject</td>
</tr>
<tr>
<td><code>setIsSelected()</code></td>
<td>(bool state=true) SimObject</td>
</tr>
<tr>
<td><code>setLocked()</code></td>
<td>(bool value=true) SimObject</td>
</tr>
<tr>
<td><code>setName()</code></td>
<td>(string newName) SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed()</code></td>
<td>(bool value=true) SimObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace()</code></td>
<td>(string name) SimObject</td>
</tr>
<tr>
<td><code>subscribe()</code></td>
<td>(SimObject listener, String event, String callback) EventManager</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>unregisterEvent()</code></td>
<td>(String event) EventManager</td>
</tr>
</tbody>
</table>
# Explosion Member List

This is the complete list of members for Explosion, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applyImpulse(Point3F pos, VectorF vel)</td>
<td>GameBase</td>
</tr>
<tr>
<td>applyRadialImpulse(Point3F origin, float radius, float magnitude)</td>
<td>GameBase</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>boundingBox</td>
<td>GameBase [static]</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dataBlock</td>
<td>GameBase</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td><code>getServerObject</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTransform</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>Explosion</code> [static]</td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>Explosion</code> [static]</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>mountNode</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>mountObject</code>(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountPID</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountPos</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountRot</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>position</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>rotation</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>save</code>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>scale</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>schedule</code>(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>scopeToClient</code>(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setCanSave</code>(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace</code>(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setControl</code>(bool controlled)</td>
<td>GameBase</td>
</tr>
<tr>
<td><code>setDataBlock</code>(GameBaseData data)</td>
<td>GameBase</td>
</tr>
<tr>
<td><code>setEditorOnly</code>(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldNamespace</code>(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue</code>(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code>(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code> setHidden</code>(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code>(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code>(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Object Type</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>setSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td>SceneObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## ExplosionData Member List

This is the complete list of members for ExplosionData, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>camShakeAmp</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>camShakeDuration</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>camShakeFalloff</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>camShakeFreq</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>camShakeRadius</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>category</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Debris</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>debrisNum</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>debrisNumVariance</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>debrisPhiMax</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>debrisPhiMin</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>debrisThetaMax</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>debrisThetaMin</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>debrisVelocity</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>debrisVelocityVariance</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class Name</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>delayMS</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>delayVariance</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>emitter</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>explosionScale</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>explosionShape</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>faceViewer</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>lifetimeMS</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>lifetimeVariance</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>lightEndBrightness</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>lightEndColor</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>lightEndRadius</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>lightNormalOffset</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>lightStartBrightness</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>lightStartColor</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>lightStartRadius</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>offset</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>onAdd(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onMount(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onNewDataBlock(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onRemove(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
</tbody>
</table>
### Function Definitions

- **onUnmount** (GameBase obj, SceneObject mountObj, int node)  
  GameBaseData

- **parentGroup**  
  SimObject

- **particleDensity**  
  ExplosionData

- **ParticleEmitter**  
  ExplosionData

- **particleRadius**  
  ExplosionData

- **persistentId**  
  SimObject

- **playSpeed**  
  ExplosionData

- **reloadOnLocalClient()**  
  SimDataBlock

- **save**(string fileName, bool selectedOnly=false, string preAppendString="")  
  SimObject

- **schedule**(float time, string method, string args...)  
  SimObject

- **setCanSave**(bool value=true)  
  SimObject

- **setClassNamespace**(string name)  
  SimObject

- **setEditorOnly**(bool value=true)  
  SimObject

- **setFieldType**(string fieldName, string type)  
  SimObject

- **setFieldValue**(string fieldName, string value, int index=-1)  
  SimObject

- **setFilename**(string fileName)  
  SimObject

- **setHidden**(bool value=true)  
  SimObject

- **setInternalName**(string newInternalName)  
  SimObject

- **setIsExpanded**(bool state=true)  
  SimObject

- **setIsSelected**(bool state=true)  
  SimObject

- **setLocked**(bool value=true)  
  SimObject

- **setName**(string newName)  
  SimObject

- **setNameChangeAllowed**(bool value=true)  
  SimObject

- **setSuperClassNamespace**(string name)  
  SimObject

- **shakeCamera**  
  ExplosionData
<table>
<thead>
<tr>
<th>sizes</th>
<th>ExplosionData</th>
</tr>
</thead>
<tbody>
<tr>
<td>soundProfile</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>subExplosion</td>
<td>ExplosionData</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>times</td>
<td>ExplosionData</td>
</tr>
</tbody>
</table>
FileDialog Member List

This is the complete list of members for `FileDialog`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>changePath</code></td>
<td>FileDialog</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>defaultFile</code></td>
<td>FileDialog</td>
</tr>
<tr>
<td><code>defaultPath</code></td>
<td>FileDialog</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>Execute()</code></td>
<td>FileDialog</td>
</tr>
<tr>
<td><code>fileName</code></td>
<td>FileDialog</td>
</tr>
<tr>
<td><code>filters</code></td>
<td>FileDialog</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>save</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>title</code></td>
<td>OpenFileDialog</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
This is the complete list of members for `FileObject`, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call()</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>close()</td>
<td>FileObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEOF()</code></td>
<td><code>FileObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>openForAppend(string filename)</code></td>
<td><code>FileObject</code></td>
</tr>
<tr>
<td><code>openForRead(string filename)</code></td>
<td><code>FileObject</code></td>
</tr>
<tr>
<td><code>openForWrite(string filename)</code></td>
<td><code>FileObject</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>peekLine()</code></td>
<td><code>FileObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>readLine()</code></td>
<td>Read a line from the file object</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td>Save the object to a file</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>Schedule an event</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td>Set whether the object can be saved</td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td>Set the class namespace</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>Set the editor-only flag</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>Set the field type</td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td>Set the field value</td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td>Set the file name</td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td>Set the hidden flag</td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td>Set the internal name</td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td>Set the is expanded flag</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td>Set the is selected flag</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>Set the locked flag</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>Set the name</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>Set whether the name change is allowed</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>Set the super class namespace</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>Get the superclass</td>
</tr>
<tr>
<td><code>writeLine(string text)</code></td>
<td>Write a line to the file object</td>
</tr>
<tr>
<td><code>writeObject(SimObject *object, string prepend)</code></td>
<td>Write the object to a file</td>
</tr>
<tr>
<td><code>writeObject(SimObject *object)</code></td>
<td>Write the object to a file</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## FileStreamObject Member List

This is the complete list of members for `FileStreamObject`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code> (SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>call</code> (string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clone</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>close</code> ()</td>
<td>FileStreamObject</td>
</tr>
<tr>
<td><code>copyFrom</code> (SimObject other)</td>
<td>StreamObject</td>
</tr>
<tr>
<td><code>deepClone</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump</code> (bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCanSave</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField</code> (int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField</code> (int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>StreamObject</code></td>
</tr>
<tr>
<td><code>getStatus()</code></td>
<td><code>StreamObject</code></td>
</tr>
<tr>
<td><code>getStreamSize()</code></td>
<td><code>StreamObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEOF()</code></td>
<td><code>StreamObject</code></td>
</tr>
<tr>
<td><code>isEOS()</code></td>
<td><code>StreamObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>open(string filename, string openMode)</code></td>
<td><code>FileStreamObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Module</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>readLine()</td>
<td>StreamObject</td>
</tr>
<tr>
<td>readLongString(int maxLength)</td>
<td>StreamObject</td>
</tr>
<tr>
<td>readString()</td>
<td>StreamObject</td>
</tr>
<tr>
<td>readSTString(bool caseSensitive=false)</td>
<td>StreamObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int newPosition)</td>
<td>StreamObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>writeLine(string line)</td>
<td>StreamObject</td>
</tr>
<tr>
<td>writeLongString(int maxLength, string string)</td>
<td>StreamObject</td>
</tr>
</tbody>
</table>
writeString(string string, int maxLength=256)  
StreamObject

Copyright © GarageGames, LLC. All Rights Reserved.
FlyingVehicle Member List

This is the complete list of members for *FlyingVehicle*, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>applyDamage(float amount)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>applyImpulse(Point3F pos, Point3F vec)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>GameBase::applyImpulse(Point3F pos, VectorF vel)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>applyRadialImpulse(Point3F origin, float radius, float magnitude)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>applyRepair(float amount)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>boundingBox</code></td>
<td><code>GameBase</code> [static]</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canCloak()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>changeMaterial(string mapTo, Material oldMat, Material newMat)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dataBlock</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>destroyThread(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>disableMove</code></td>
<td><code>Vehicle</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>dump( bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMeshVisibility()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getAIRepairPoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getControllingClient()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getControllingObject()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageFlash()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamagePercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageState()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDefaultValue</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEnergyLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEnergyPercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getEyePoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeTransform()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>getEyeVector()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getImageAltTrigger(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageAmmo(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageGenericTrigger(int slot, int trigger)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageLoaded(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageScriptAnimPrefix(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageSkinTag(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageState(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageTarget(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageTrigger(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getLookAtPoint(float distance=2000, int typeMask=0xFFFFFFFF)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMaxDamage()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getModelFile()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMountedImage(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountSlot(ShapeBaseImageData)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzlePoint(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzleVector(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPendingImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRechargeRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRepairRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getShapeName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSkinName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSlotTransform(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTargetCount()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTargetName(int index)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getVelocity()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getWhiteOut()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hasImageState(int slot, string state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isAIControlled</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isCloaked()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isDestroyed()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isDisabled()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEnabled()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isHidden()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isImageFiring(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isImageMounted(ShapeBaseImageData image)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>FlyingVehicle [static]</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>FlyingVehicle [static]</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountImage(ShapeBaseImageData, int slot, bool loaded=true, string skinTag=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pauseThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAudio(int slot, SFXTrack track)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>playThread(int slot, string name=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setAllMeshesHidden(bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class(s)</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCloaked(bool cloak)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setControl(bool controlled)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setDamageFlash(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageLevel(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageState(string state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageVector(Point3F vec)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDataBlock(GameBaseData data)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEnergyLevel(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool show)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAltTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAmmo(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageGenericTrigger(int slot, int trigger, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageLoaded(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageScriptAnimPrefix(int slot, string prefix)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTarget(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInvincibleMode(float time, float speed)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setMeshHidden(string name, bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setRechargeRate(float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setRepairRate(float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setShapeName(string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setSkinName(string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setThreadDir(int slot, bool fwd)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setThreadPosition(int slot, float pos)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setThreadTimeScale(int slot, float scale)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setVelocity(Point3F vel)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setWhiteOut(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>skin</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>startFade(int time, int delay, bool fadeOut)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>stopAudio(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>stopThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>useCreateHeight(bool enabled)</td>
<td>FlyingVehicle</td>
</tr>
<tr>
<td>validateCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
</tbody>
</table>
| workingQueryBoxSizeMultiplier             | Vehicle          | [static]
| workingQueryBoxStaleThreshold | Vehicle | [static] |

Copyright © GarageGames, LLC. All Rights Reserved.
# FlyingVehicleData Member List

This is the complete list of members for *FlyingVehicleData*, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>autoAngularForce</code></td>
<td><code>FlyingVehicleData</code></td>
</tr>
<tr>
<td><code>autoInputDamping</code></td>
<td><code>FlyingVehicleData</code></td>
</tr>
<tr>
<td><code>autoLinearForce</code></td>
<td><code>FlyingVehicleData</code></td>
</tr>
<tr>
<td><code>backwardJetEmitter</code></td>
<td><code>FlyingVehicleData</code></td>
</tr>
<tr>
<td><code>bodyFriction</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>bodyRestitution</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>cameraDecay</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>cameraDefaultFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraLag</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>cameraMaxDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMaxFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraOffset</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>cameraRoll</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>category</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>checkDeployPos(TransformF tfm)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>collDamageMultiplier</td>
<td>VehicleData</td>
</tr>
<tr>
<td>collDamageThresholdVel</td>
<td>VehicleData</td>
</tr>
<tr>
<td>collisionTol</td>
<td>VehicleData</td>
</tr>
<tr>
<td>computeCRC</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>contactTol</td>
<td>VehicleData</td>
</tr>
<tr>
<td>createHoverHeight</td>
<td>FlyingVehicleData</td>
</tr>
<tr>
<td>cubeReflectorDesc</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>damageEmitter</td>
<td>VehicleData</td>
</tr>
<tr>
<td>damageEmitterOffset</td>
<td>VehicleData</td>
</tr>
<tr>
<td>damageLevelTolerance</td>
<td>VehicleData</td>
</tr>
<tr>
<td>Debris</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>debrisShapeName</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>density</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>destroyedLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>disabledLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>downJetEmitter</td>
<td>FlyingVehicleData</td>
</tr>
<tr>
<td>drag</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dustEmitter</td>
<td>VehicleData</td>
</tr>
<tr>
<td>dustHeight</td>
<td>VehicleData</td>
</tr>
<tr>
<td>engineSound</td>
<td>FlyingVehicleData</td>
</tr>
<tr>
<td>exitingWater</td>
<td>VehicleData</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>exitSplashSoundVelocity</td>
<td>VehicleData</td>
</tr>
<tr>
<td>Explosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>firstPersonOnly</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>forwardJetEmitter</td>
<td>FlyingVehicleData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClass()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeployTransform(Point3F pos, Point3F normal)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hardImpactSound</td>
<td>VehicleData</td>
</tr>
<tr>
<td>hardImpactSpeed</td>
<td>VehicleData</td>
</tr>
<tr>
<td>hardSplashSoundVelocity</td>
<td>VehicleData</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizontalSurfaceForce</td>
<td>FlyingVehicleData</td>
</tr>
<tr>
<td>Variable</td>
<td>Data Type</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>hoverHeight</td>
<td>FlyingVehicleData</td>
</tr>
<tr>
<td>impactWaterEasy</td>
<td>VehicleData</td>
</tr>
<tr>
<td>impactWaterHard</td>
<td>VehicleData</td>
</tr>
<tr>
<td>impactWaterMedium</td>
<td>VehicleData</td>
</tr>
<tr>
<td>inheritEnergyFromMount</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>integration</td>
<td>VehicleData</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup( SimGroup group )</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInvincible</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>jetEnergyDrain</td>
<td>VehicleData</td>
</tr>
<tr>
<td>jetForce</td>
<td>VehicleData</td>
</tr>
<tr>
<td>jetSound</td>
<td>FlyingVehicleData</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>maneuveringForce</td>
<td>FlyingVehicleData</td>
</tr>
<tr>
<td>mass</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>massBox</td>
<td>VehicleData</td>
</tr>
<tr>
<td>massCenter</td>
<td>VehicleData</td>
</tr>
<tr>
<td>maxAutoSpeed</td>
<td>FlyingVehicleData</td>
</tr>
<tr>
<td>maxDamage</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>maxDrag</td>
<td>VehicleData</td>
</tr>
<tr>
<td>Property</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td><code>maxEnergy</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>maxSteeringAngle</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>mediumSplashSoundVelocity</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>minDrag</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>minImpactSpeed</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>minJetEnergy</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>minRollSpeed</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>minTrailSpeed</code></td>
<td><code>FlyingVehicleData</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>numDmgEmitterAreas</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>observeThroughObject</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onAdd</code> (GameBase obj)</td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onCollision</code> (ShapeBase obj, SceneObject collObj, VectorF vec, float len)</td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onDamage</code> (ShapeBase obj, float delta)</td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onDestroyed</code> (ShapeBase obj, string lastState)</td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onDisabled</code> (ShapeBase obj, string lastState)</td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onEnabled</code> (ShapeBase obj, string lastState)</td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onEndSequence</code> (ShapeBase obj, int slot)</td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onEnterLiquid</code> (Vehicle obj, float coverage, string type)</td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>onForceUncloak</code> (ShapeBase obj, string reason)</td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onImpact</code> (ShapeBase obj, SceneObject collObj, VectorF vec, float len)</td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onLeaveLiquid</code> (Vehicle obj, string type)</td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>onMount</code> (GameBase obj, SceneObject mountObj, int node)</td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onNewDataBlock</code> (GameBase obj)</td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td><code>onRemove</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onTrigger</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onUnmount</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>reloadOnLocalClient</code></td>
<td><code>SimDataBlock</code></td>
</tr>
<tr>
<td><code>renderWhenDestroyed</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>repairRate</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>rollForce</code></td>
<td><code>FlyingVehicleData</code></td>
</tr>
<tr>
<td><code>rotationalDrag</code></td>
<td><code>FlyingVehicleData</code></td>
</tr>
<tr>
<td><code>save</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>shadowEnable</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowMaxVisibleDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowProjectionDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSize</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSphereAdjust</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shapeFile</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>softImpactSound</td>
<td>VehicleData</td>
</tr>
<tr>
<td>softImpactSpeed</td>
<td>VehicleData</td>
</tr>
<tr>
<td>softSplashSoundVelocity</td>
<td>VehicleData</td>
</tr>
<tr>
<td>splashEmitter</td>
<td>VehicleData</td>
</tr>
<tr>
<td>splashFreqMod</td>
<td>VehicleData</td>
</tr>
<tr>
<td>splashVelEpsilon</td>
<td>VehicleData</td>
</tr>
<tr>
<td>steeringForce</td>
<td>FlyingVehicleData</td>
</tr>
<tr>
<td>steeringRollForce</td>
<td>FlyingVehicleData</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>trailEmitter</td>
<td>FlyingVehicleData</td>
</tr>
<tr>
<td>triggerDustHeight</td>
<td>VehicleData</td>
</tr>
<tr>
<td>underwaterExplosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>useEyePoint</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>verticalSurfaceForce</td>
<td>FlyingVehicleData</td>
</tr>
<tr>
<td>vertThrustMultiple</td>
<td>FlyingVehicleData</td>
</tr>
<tr>
<td>waterWakeSound</td>
<td>VehicleData</td>
</tr>
</tbody>
</table>
Forest Member List

This is the complete list of members for Forest, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clear()</code></td>
<td>Forest</td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dataFile</code></td>
<td>Forest</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>disableImposters</code></td>
<td>Forest [static]</td>
</tr>
<tr>
<td><code>drawBounds</code></td>
<td>Forest [static]</td>
</tr>
<tr>
<td><code>drawCells</code></td>
<td>Forest [static]</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>forceImposters</code></td>
<td>Forest [static]</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isDirty()</code></td>
<td><code>Forest</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>Forest</code> [static]</td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>Forest</code> [static]</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>lodReflectScalar</td>
<td>Forest</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>regenCells()</td>
<td>Forest</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>saveDataFile</td>
<td>Forest</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setScale</code> (Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setScopeAlways</code> ()</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setTransform</code> (TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>unmount</code> ()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>unmountObject</code> (SceneObject target)</td>
<td>SceneObject</td>
</tr>
</tbody>
</table>
## ForestBrushElement Member List

This is the complete list of members for `ForestBrushElement`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>elevationMax</td>
<td>ForestBrushElement</td>
</tr>
<tr>
<td>elevationMin</td>
<td>ForestBrushElement</td>
</tr>
<tr>
<td>ForestItemData</td>
<td>ForestBrushElement</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>probability</code></td>
<td><code>ForestBrushElement</code></td>
</tr>
<tr>
<td><code>rotationRange</code></td>
<td><code>ForestBrushElement</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool</code></td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>selectedOnly=false, string preAppendString=&quot;&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scaleExponent</td>
<td>ForestBrushElement</td>
</tr>
<tr>
<td>scaleMax</td>
<td>ForestBrushElement</td>
</tr>
<tr>
<td>scaleMin</td>
<td>ForestBrushElement</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>sinkMax</td>
<td>ForestBrushElement</td>
</tr>
<tr>
<td>sinkMin</td>
<td>ForestBrushElement</td>
</tr>
<tr>
<td>sinkRadius</td>
<td>ForestBrushElement</td>
</tr>
<tr>
<td>slopeMax</td>
<td>ForestBrushElement</td>
</tr>
<tr>
<td>slopeMin</td>
<td>ForestBrushElement</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>
# ForestItemData Member List

This is the complete list of members for `ForestItemData`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>branchAmp</code></td>
<td><code>ForestItemData</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>collidable</code></td>
<td><code>ForestItemData</code></td>
</tr>
<tr>
<td><code>dampingCoefficient</code></td>
<td><code>ForestItemData</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>detailAmp</code></td>
<td><code>ForestItemData</code></td>
</tr>
<tr>
<td><code>detailFreq</code></td>
<td><code>ForestItemData</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getld()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mass</td>
<td>ForestItemData</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentld</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class/Simulation Object</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>radius</td>
<td>ForestItemData</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>rigidity</td>
<td>ForestItemData</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassName(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>shapeFile</td>
<td>ForestItemData</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tightnessCoefficient</td>
<td>ForestItemData</td>
</tr>
<tr>
<td>trunkBendScale</td>
<td>ForestItemData</td>
</tr>
<tr>
<td>windScale</td>
<td>ForestItemData</td>
</tr>
</tbody>
</table>
This is the complete list of members for **ForestWindEmitter**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code> (SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>attachToObject</code> (int objectID)</td>
<td>ForestWindEmitter</td>
</tr>
<tr>
<td><code>call</code> (string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient</code> (NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump</code> (bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getEulerRotation()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getScale()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>gustFrequency</td>
<td>ForestWindEmitter</td>
</tr>
<tr>
<td>gustStrength</td>
<td>ForestWindEmitter</td>
</tr>
<tr>
<td>gustWobbleStrength</td>
<td>ForestWindEmitter</td>
</tr>
<tr>
<td>gustYawAngle</td>
<td>ForestWindEmitter</td>
</tr>
<tr>
<td>gustYawFrequency</td>
<td>ForestWindEmitter</td>
</tr>
<tr>
<td>hasMount</td>
<td>ForestWindEmitter</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>ForestWindEmitter</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>ForestWindEmitter</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function/Property</td>
<td>Class/Type</td>
</tr>
<tr>
<td>------------------------------------------------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>radialEmitter</td>
<td>ForestWindEmitter</td>
</tr>
<tr>
<td>radius</td>
<td>ForestWindEmitter</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class Name</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>strength</td>
<td>ForestWindEmitter</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>turbulenceFrequency</td>
<td>ForestWindEmitter</td>
</tr>
<tr>
<td>turbulenceStrength</td>
<td>ForestWindEmitter</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>windEnabled</td>
<td>ForestWindEmitter</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## fxFoliageReplicator Member List

This is the complete list of members for fxFoliageReplicator, including all inherited members.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AllowedTerrainSlope</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>AllowOnInteriors</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>AllowOnStatics</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>AllowOnTerrain</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>AllowOnWater</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>AllowWaterSurface</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>AlphaCutoff</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>assignFieldsFrom</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>CullResolution</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>DebugBoxHeight</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method/Property</td>
<td>Class/Type</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>FadeInRegion</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>FadeOutRegion</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>FixAspectRatio</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>FixSizeToMax</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>FoliageCount</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>FoliageFile</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>FoliageRetries</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>GroundAlpha</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>HideFoliage</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>InnerRadiusX</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>InnerRadiusY</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>LightOn</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>LightSync</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>lightTime</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>MaxHeight</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>MaxLuminance</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>MaxSwayTime</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>MaxWidth</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>MinHeight</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>MinLuminance</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>MinSwayTime</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>MinWidth</td>
<td>fxFoliageReplicator</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function/Property</td>
<td>Class/Type</td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td><code>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountPID</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountPos</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountRot</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>OffsetZ</code></td>
<td><code>fxFoliageReplicator</code></td>
</tr>
<tr>
<td><code>OuterRadiusX</code></td>
<td><code>fxFoliageReplicator</code></td>
</tr>
<tr>
<td><code>OuterRadiusY</code></td>
<td><code>fxFoliageReplicator</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>PlacementAreaHeight</code></td>
<td><code>fxFoliageReplicator</code></td>
</tr>
<tr>
<td><code>PlacementColour</code></td>
<td><code>fxFoliageReplicator</code></td>
</tr>
<tr>
<td><code>position</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>RandomFlip</code></td>
<td><code>fxFoliageReplicator</code></td>
</tr>
<tr>
<td><code>rotation</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>scale</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>scopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>seed</code></td>
<td><code>fxFoliageReplicator</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setScale</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>ShowPlacementArea</code></td>
<td><code>fxFoliageReplicator</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>SwayMagFront</code></td>
<td><code>fxFoliageReplicator</code></td>
</tr>
<tr>
<td><code>SwayMagSide</code></td>
<td><code>fxFoliageReplicator</code></td>
</tr>
<tr>
<td><code>SwayOn</code></td>
<td><code>fxFoliageReplicator</code></td>
</tr>
<tr>
<td><code>SwaySync</code></td>
<td><code>fxFoliageReplicator</code></td>
</tr>
<tr>
<td><code>unmount</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountObject</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>UseCulling</code></td>
<td><code>fxFoliageReplicator</code></td>
</tr>
<tr>
<td><code>UseDebugInfo</code></td>
<td><code>fxFoliageReplicator</code></td>
</tr>
<tr>
<td><code>useTrueBillboards</code></td>
<td><code>fxFoliageReplicator</code></td>
</tr>
<tr>
<td><code>ViewClosest</code></td>
<td><code>fxFoliageReplicator</code></td>
</tr>
</tbody>
</table>
Copyright © GarageGames, LLC. All Rights Reserved.
# fxShapeReplicatedStatic Member List

This is the complete list of members for `fxShapeReplicatedStatic`, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allowPlayerStep</td>
<td><code>fxShapeReplicatedStatic</code></td>
</tr>
<tr>
<td>assignFieldsFrom</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>assignPersistentId</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>call</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>canSave</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>class</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>className</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>clearScopeToClient</td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td>clone</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>collisionType</td>
<td><code>fxShapeReplicatedStatic</code></td>
</tr>
<tr>
<td>decalType</td>
<td><code>fxShapeReplicatedStatic</code></td>
</tr>
<tr>
<td>deepClone</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>delete</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dump</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpClassHierarchy</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpGroupHierarchy</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpMethods</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>forceDetail</td>
<td><code>fxShapeReplicatedStatic</code></td>
</tr>
<tr>
<td>getCanSave</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>getClassName</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>getClassNamespace</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getEulerRotation()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getScale()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method/Property</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>fxShapeReplicatedStatic</code> [static]</td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>fxShapeReplicatedStatic</code> [static]</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>locked</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>meshCulling</td>
<td><code>fxShapeReplicatedStatic</code></td>
</tr>
<tr>
<td>mountNode</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountObject</code> (SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>mountPID</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>mountPos</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>mountRot</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>name</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>originSort</td>
<td><code>fxShapeReplicatedStatic</code></td>
</tr>
<tr>
<td>parentGroup</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>persistentId</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>playAmbient</td>
<td><code>fxShapeReplicatedStatic</code></td>
</tr>
<tr>
<td>position</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>renderNormals</td>
<td><code>fxShapeReplicatedStatic</code></td>
</tr>
<tr>
<td>rotation</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>save</code> (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>scale</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>schedule</code> (float time, string method, string args...)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>scopeToClient</code> (NetConnection client)</td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setCanSave</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method Call</td>
<td>Return Type</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setScale</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>shapeName</code></td>
<td><code>fxShapeReplicatedStatic</code></td>
</tr>
<tr>
<td><code>skin</code></td>
<td><code>fxShapeReplicatedStatic</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>unmount</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountObject</code></td>
<td><code>SceneObject</code></td>
</tr>
</tbody>
</table>
### fxShapeReplicator Member List

This is the complete list of members for `fxShapeReplicator`, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlignToTerrain</td>
<td><code>fxShapeReplicator</code></td>
</tr>
<tr>
<td>AllowedTerrainSlope</td>
<td><code>fxShapeReplicator</code></td>
</tr>
<tr>
<td>AllowOnInteriors</td>
<td><code>fxShapeReplicator</code></td>
</tr>
<tr>
<td>AllowOnStatics</td>
<td><code>fxShapeReplicator</code></td>
</tr>
<tr>
<td>AllowOnTerrain</td>
<td><code>fxShapeReplicator</code></td>
</tr>
<tr>
<td>AllowOnWater</td>
<td><code>fxShapeReplicator</code></td>
</tr>
<tr>
<td>AllowWaterSurface</td>
<td><code>fxShapeReplicator</code></td>
</tr>
<tr>
<td>assignFieldsFrom</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>assignPersistentId</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>call</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>canSave</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>class</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>className</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>clearScopeToClient</td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td>clone</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>deepClone</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>delete</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dump</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpClassHierarchy</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpGroupHierarchy</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpMethods</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getEulerRotation()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getID()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObjectType()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>hideReplications</td>
<td>fxShapeReplicator</td>
</tr>
<tr>
<td>InnerRadiusX</td>
<td>fxShapeReplicator</td>
</tr>
<tr>
<td>InnerRadiusY</td>
<td>fxShapeReplicator</td>
</tr>
<tr>
<td>Interactions</td>
<td>fxShapeReplicator</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>fxShapeReplicator [static]</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>fxShapeReplicator [static]</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>OffsetZ</td>
<td>fxShapeReplicator</td>
</tr>
<tr>
<td>OuterRadiusX</td>
<td>fxShapeReplicator</td>
</tr>
<tr>
<td>OuterRadiusY</td>
<td>fxShapeReplicator</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>PlacementAreaHeight</td>
<td>fxShapeReplicator</td>
</tr>
<tr>
<td>PlacementColour</td>
<td>fxShapeReplicator</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method,</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>scopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>seed</code></td>
<td><code>fxShapeReplicator</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>ShapeCount</code></td>
<td><code>fxShapeReplicator</code></td>
</tr>
<tr>
<td><code>shapeFile</code></td>
<td><code>fxShapeReplicator</code></td>
</tr>
<tr>
<td><code>ShapeRetries</code></td>
<td><code>fxShapeReplicator</code></td>
</tr>
<tr>
<td><code>ShapeRotateMax</code></td>
<td><code>fxShapeReplicator</code></td>
</tr>
<tr>
<td><strong>ShapeRotateMin</strong></td>
<td><strong>fxShapeReplicator</strong></td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td><strong>ShapeScaleMax</strong></td>
<td><strong>fxShapeReplicator</strong></td>
</tr>
<tr>
<td><strong>ShapeScaleMin</strong></td>
<td><strong>fxShapeReplicator</strong></td>
</tr>
<tr>
<td><strong>ShowPlacementArea</strong></td>
<td><strong>fxShapeReplicator</strong></td>
</tr>
<tr>
<td><strong>superClass</strong></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><strong>TerrainAlignment</strong></td>
<td><strong>fxShapeReplicator</strong></td>
</tr>
<tr>
<td><strong>unmount()</strong></td>
<td><strong>SceneObject</strong></td>
</tr>
<tr>
<td><strong>unmountObject(SceneObject target)</strong></td>
<td><strong>SceneObject</strong></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## GameBase Member List

This is the complete list of members for GameBase, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>applyImpulse(Point3F pos, VectorF vel)</td>
<td>GameBase</td>
</tr>
<tr>
<td>applyRadialImpulse(Point3F origin, float radius, float magnitude)</td>
<td>GameBase</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>boundingBox</td>
<td>GameBase</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dataBlock</td>
<td>GameBase</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>GameBase</code>     [static]</td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>GameBase</code>     [static]</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>mountNode</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Category</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>mountObject(SeeObject objB, int slot,</td>
<td>SceneObject</td>
</tr>
<tr>
<td>TransformF txfm=MatrixF::Identity)</td>
<td></td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setControl(bool controlled)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setDataBlock(GameBaseData data)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td><code>setSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td>SceneObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# GameBaseData Member List

This is the complete list of members for **GameBaseData**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>category</code></td>
<td>GameBaseData</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Function Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>getFieldType</code> (string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue</code> (string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup</code> (SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField</code> (string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass</code> (string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod</code> (string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>onAdd</code> (GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td><code>onMount</code> (GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td><code>onNewDataBlock</code> (GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td><code>onRemove</code> (GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td><code>onUnmount</code> (GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>
# GameConnection Member List

This is the complete list of members for GameConnection, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>acceptsAsChild(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>activateGhosting()</code></td>
<td>GameConnection</td>
</tr>
<tr>
<td><code>add(SimObject objects...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>bringToFront(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>callOnChildren(string method, string args...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>callOnChildrenNoRecurse(string method, string args...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>chaseCam(int size)</code></td>
<td>GameConnection</td>
</tr>
<tr>
<td><code>checkMaxRate()</code></td>
<td>NetConnection</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clear()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>clearCameraObject()</code></td>
<td>GameConnection</td>
</tr>
<tr>
<td><code>clearPaths()</code></td>
<td>NetConnection</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>connect(string remoteAddress)</code></td>
<td>NetConnection</td>
</tr>
<tr>
<td><code>connectLocal()</code></td>
<td>NetConnection</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class Name</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>delete (string reason=&quot;&quot;)</td>
<td>GameConnection</td>
</tr>
<tr>
<td>NetConnection::delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump (bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>findObjectByInternalName (string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAddress()</td>
<td>NetConnection</td>
</tr>
<tr>
<td>getCameraObject()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getControlCameraDefaultFov()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>getControlCameraFov()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>getControlObject()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDamageFlash()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField (int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField (int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType (string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue (string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGhostID(int realID)</td>
<td>NetConnection</td>
</tr>
<tr>
<td>getGhostsActive()</td>
<td>NetConnection</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getPacketLoss()</td>
<td>NetConnection</td>
</tr>
<tr>
<td>getPing()</td>
<td>NetConnection</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getServerConnection()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getWhiteOut()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>initialControlSet()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isAIControlled()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isControlObjectRotDampedCamera()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>isDemoPlaying()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>isDemoRecording()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstPerson()</td>
<td>GameConnection</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>isInNamespaceHierarchy</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>listClassIDs</td>
<td>GameConnection</td>
</tr>
<tr>
<td>listObjects</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onConnectionAccepted</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onConnectionDropped</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onConnectionError</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onConnectionTimedOut</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onConnectRequestRejected</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onConnectRequestTimedOut</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onControlObjectChange</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onDataBlocksDone</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onDrop</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onFlash</td>
<td>GameConnection</td>
</tr>
<tr>
<td>onObjectAdded</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved</td>
<td>SimSet</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>play2D</td>
<td>GameConnection</td>
</tr>
<tr>
<td>play3D</td>
<td>GameConnection</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>playDemo</td>
<td>GameConnection</td>
</tr>
<tr>
<td>pushToBack</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild</td>
<td>SimSet</td>
</tr>
<tr>
<td>resetGhosting</td>
<td>GameConnection</td>
</tr>
<tr>
<td>resolveGhostID</td>
<td>NetConnection</td>
</tr>
<tr>
<td>resolveObjectFromGhostIndex</td>
<td>NetConnection</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>setBlackOut</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setCameraObject</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setConnectArgs</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setControlCameraFov</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setControlObject</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstPerson</td>
<td>GameConnection</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setJoinPassword</code> (string password)</td>
<td>GameConnection</td>
</tr>
<tr>
<td><code>setLagIcon</code> (bool state)</td>
<td>GameConnection</td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setMissionCRC</code> (int CRC)</td>
<td>GameConnection</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setSimulatedNetParams</code> (float packetLoss, int delay)</td>
<td>NetConnection</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>sort</code> (string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>startRecording</code> (string fileName)</td>
<td>GameConnection</td>
</tr>
<tr>
<td><code>stopRecording</code> ()</td>
<td>GameConnection</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>transmitDataBlocks</code> (int sequence)</td>
<td>GameConnection</td>
</tr>
<tr>
<td><code>transmitPaths</code> ()</td>
<td>NetConnection</td>
</tr>
</tbody>
</table>
# GameTSCtrl Member List

This is the complete list of members for GameTSCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>anchorBottom</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorLeft</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorRight</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorTop</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>calculateViewDistance(float radius)</td>
<td>GuiTSCtrl</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>cameraZRot</td>
<td>GuiTSCtrl</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>Function/Method</td>
<td>Class/Type</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td><code>clearFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>command</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>controlsChild(GuiControl control)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>docking</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>extent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControl(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControls(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findObjectByInternalName(string internalName, bool searchChildren=false)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>forceFOV</code></td>
<td><code>GuiTSCtrl</code></td>
</tr>
<tr>
<td><code>getAspect()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getWorldToScreenScale()</td>
<td>GuiTSCtrl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>dropPoint)</td>
<td></td>
</tr>
<tr>
<td><code>onControlDragExit(GuiControl control, Point2I dropPoint)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onControlDragged(GuiControl control, Point2I dropPoint)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onControlDropped(GuiControl control, Point2I dropPoint)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onDialogPop()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onDialogPush()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onGainFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onLoseFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onObjectAdded(SimObject object)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onObjectRemoved(SimObject object)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onRemove()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onSleep()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onVisible(bool state)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onWake()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>padding</code></td>
<td>GuiContainer</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>pointInControl(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>position</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>profile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>project(Point3F worldPosition)</code></td>
<td>GuiTSCtrl</td>
</tr>
<tr>
<td><code>pushToBack(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>reflectPriority</code></td>
<td>GuiTSCtrl</td>
</tr>
<tr>
<td><code>remove(SimObject objects...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>reorderChild(SimObject child1, SimObject child2)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>unproject(Point3F screenPosition)</td>
<td>GuiTSCtrl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
## GFXCardProfilerAPI Member List

This is the complete list of members for **GFXCardProfilerAPI**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>getCard()</code></td>
<td>GFXCardProfilerAPI</td>
<td>[static]</td>
</tr>
<tr>
<td><code>getRenderer()</code></td>
<td>GFXCardProfilerAPI</td>
<td>[static]</td>
</tr>
<tr>
<td><code>getVendor()</code></td>
<td>GFXCardProfilerAPI</td>
<td>[static]</td>
</tr>
<tr>
<td><code>getVersion()</code></td>
<td>GFXCardProfilerAPI</td>
<td>[static]</td>
</tr>
<tr>
<td><code>getVideoMemoryMB()</code></td>
<td>GFXCardProfilerAPI</td>
<td>[static]</td>
</tr>
<tr>
<td><code>queryProfile</code> (string name, int defaultValue)`</td>
<td>GFXCardProfilerAPI</td>
<td>[static]</td>
</tr>
<tr>
<td><code>setCapability</code> (string name, int value)`</td>
<td>GFXCardProfilerAPI</td>
<td>[static]</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# GFXInit Member List

This is the complete list of members for `GFXInit`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>createNullDevice()</code></td>
<td><code>GFXInit</code> [static]</td>
</tr>
<tr>
<td><code>getAdapterCount()</code></td>
<td><code>GFXInit</code> [static]</td>
</tr>
<tr>
<td><code>getAdapterMode(int index, int modeIndex)</code></td>
<td><code>GFXInit</code> [static]</td>
</tr>
<tr>
<td><code>getAdapterModeCount(int index)</code></td>
<td><code>GFXInit</code> [static]</td>
</tr>
<tr>
<td><code>getAdapterName(int index)</code></td>
<td><code>GFXInit</code> [static]</td>
</tr>
<tr>
<td><code>getAdapterShaderModel(int index)</code></td>
<td><code>GFXInit</code> [static]</td>
</tr>
<tr>
<td><code>getAdapterType(int index)</code></td>
<td><code>GFXInit</code> [static]</td>
</tr>
<tr>
<td><code>getDefaultAdapterIndex()</code></td>
<td><code>GFXInit</code> [static]</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# GFXSamplerStateData Member List

This is the complete list of members for `GFXSamplerStateData`, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>addressModeU</td>
<td><code>GFXSamplerStateData</code></td>
</tr>
<tr>
<td>addressModeV</td>
<td><code>GFXSamplerStateData</code></td>
</tr>
<tr>
<td>addressModeW</td>
<td><code>GFXSamplerStateData</code></td>
</tr>
<tr>
<td>alphaArg1</td>
<td><code>GFXSamplerStateData</code></td>
</tr>
<tr>
<td>alphaArg2</td>
<td><code>GFXSamplerStateData</code></td>
</tr>
<tr>
<td>alphaArg3</td>
<td><code>GFXSamplerStateData</code></td>
</tr>
<tr>
<td>alphaOp</td>
<td><code>GFXSamplerStateData</code></td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>canSave</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>class</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>className</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>clone()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>colorArg1</td>
<td><code>GFXSamplerStateData</code></td>
</tr>
<tr>
<td>colorArg2</td>
<td><code>GFXSamplerStateData</code></td>
</tr>
<tr>
<td>colorArg3</td>
<td><code>GFXSamplerStateData</code></td>
</tr>
<tr>
<td>deepClone()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>delete()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpMethods()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>magFilter</td>
<td>GFXSamplerStateData</td>
</tr>
<tr>
<td>maxAnisotropy</td>
<td>GFXSamplerStateData</td>
</tr>
<tr>
<td>minFilter</td>
<td>GFXSamplerStateData</td>
</tr>
<tr>
<td>mipFilter</td>
<td>GFXSamplerStateData</td>
</tr>
<tr>
<td>mipLODBias</td>
<td>GFXSamplerStateData</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>resultArg</td>
<td>GFXSamplerStateData</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>textureColorOp</code></td>
<td><code>GFXSamplerStateData</code></td>
</tr>
<tr>
<td><code>textureTransform</code></td>
<td><code>GFXSamplerStateData</code></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## GFXStateBlockData Member List

This is the complete list of members for GFXStateBlockData, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>alphaDefined</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>alphaTestEnable</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>alphaTestFunc</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>alphaTestRef</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>blendDefined</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>blendDest</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>blendEnable</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>blendOp</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>blendSrc</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>colorWriteAlpha</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>colorWriteBlue</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>colorWriteDefined</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>colorWriteGreen</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>colorWriteRed</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>cullDefined</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>cullMode</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>ffLighting</code></td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>samplersDefined</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>samplerStates</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>separateAlphaBlendDefined</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>separateAlphaBlendDest</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>separateAlphaBlendEnable</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>separateAlphaBlendOp</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>separateAlphaBlendSrc</td>
<td>GFXStateBlockData</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>stencilDefined</code></td>
<td><code>GFXStateBlockData</code></td>
</tr>
<tr>
<td><code>stencilEnable</code></td>
<td><code>GFXStateBlockData</code></td>
</tr>
<tr>
<td><code>stencilFailOp</code></td>
<td><code>GFXStateBlockData</code></td>
</tr>
<tr>
<td><code>stencilFunc</code></td>
<td><code>GFXStateBlockData</code></td>
</tr>
<tr>
<td><code>stencilMask</code></td>
<td><code>GFXStateBlockData</code></td>
</tr>
<tr>
<td><code>stencilPassOp</code></td>
<td><code>GFXStateBlockData</code></td>
</tr>
<tr>
<td><code>stencilRef</code></td>
<td><code>GFXStateBlockData</code></td>
</tr>
<tr>
<td><code>stencilWriteMask</code></td>
<td><code>GFXStateBlockData</code></td>
</tr>
<tr>
<td><code>stencilZFailOp</code></td>
<td><code>GFXStateBlockData</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>textureFactor</code></td>
<td><code>GFXStateBlockData</code></td>
</tr>
<tr>
<td><code>vertexColorEnable</code></td>
<td><code>GFXStateBlockData</code></td>
</tr>
<tr>
<td><code>zBias</code></td>
<td><code>GFXStateBlockData</code></td>
</tr>
<tr>
<td><code>zDefined</code></td>
<td><code>GFXStateBlockData</code></td>
</tr>
<tr>
<td><code>zEnable</code></td>
<td><code>GFXStateBlockData</code></td>
</tr>
<tr>
<td><code>zFunc</code></td>
<td><code>GFXStateBlockData</code></td>
</tr>
<tr>
<td><code>zSlopeBias</code></td>
<td><code>GFXStateBlockData</code></td>
</tr>
<tr>
<td><code>zWriteEnable</code></td>
<td><code>GFXStateBlockData</code></td>
</tr>
</tbody>
</table>
# GroundCover Member List

This is the complete list of members for `GroundCover`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>billboardUVs</code></td>
<td><code>GroundCover</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clumpExponent</code></td>
<td><code>GroundCover</code></td>
</tr>
<tr>
<td><code>clumpRadius</code></td>
<td><code>GroundCover</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dissolveRadius</code></td>
<td><code>GroundCover</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getEulerRotation()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getScale()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>gridSize</code></td>
<td><code>GroundCover</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>invertLayer</code></td>
<td><code>GroundCover</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>GroundCover</code> [static]</td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>GroundCover</code> [static]</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>layer</code></td>
<td><code>GroundCover</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>lockFrustum</td>
<td>GroundCover</td>
</tr>
<tr>
<td>Material</td>
<td>GroundCover</td>
</tr>
<tr>
<td>maxBillboardTiltAngle</td>
<td>GroundCover</td>
</tr>
<tr>
<td>maxClumpCount</td>
<td>GroundCover</td>
</tr>
<tr>
<td>maxElements</td>
<td>GroundCover</td>
</tr>
<tr>
<td>maxElevation</td>
<td>GroundCover</td>
</tr>
<tr>
<td>maxSlope</td>
<td>GroundCover</td>
</tr>
<tr>
<td>minClumpCount</td>
<td>GroundCover</td>
</tr>
<tr>
<td>minElevation</td>
<td>GroundCover</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>noBillboards</td>
<td>GroundCover</td>
</tr>
<tr>
<td>noShapes</td>
<td>GroundCover</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>probability</td>
<td>GroundCover</td>
</tr>
<tr>
<td>radius</td>
<td>GroundCover</td>
</tr>
<tr>
<td>reflectScale</td>
<td>GroundCover</td>
</tr>
<tr>
<td>renderCells</td>
<td>GroundCover</td>
</tr>
<tr>
<td>renderedBatches</td>
<td>GroundCover [static]</td>
</tr>
<tr>
<td>renderedBillboards</td>
<td>GroundCover [static]</td>
</tr>
<tr>
<td>renderedCells</td>
<td>GroundCover [static]</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>renderedShapes</td>
<td>GroundCover [static]</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>seed</td>
<td>GroundCover</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>shapeCullRadius</td>
<td>GroundCover</td>
</tr>
<tr>
<td>Property</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>shapeFilename</code></td>
<td>GroundCover</td>
</tr>
<tr>
<td><code>shapesCastShadows</code></td>
<td>GroundCover</td>
</tr>
<tr>
<td><code>sizeExponent</code></td>
<td>GroundCover</td>
</tr>
<tr>
<td><code>sizeMax</code></td>
<td>GroundCover</td>
</tr>
<tr>
<td><code>sizeMin</code></td>
<td>GroundCover</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>windDirection</code></td>
<td>GroundCover</td>
</tr>
<tr>
<td><code>windGustFrequency</code></td>
<td>GroundCover</td>
</tr>
<tr>
<td><code>windGustLength</code></td>
<td>GroundCover</td>
</tr>
<tr>
<td><code>windGustStrength</code></td>
<td>GroundCover</td>
</tr>
<tr>
<td><code>windScale</code></td>
<td>GroundCover</td>
</tr>
<tr>
<td><code>windTurbulenceFrequency</code></td>
<td>GroundCover</td>
</tr>
<tr>
<td><code>windTurbulenceStrength</code></td>
<td>GroundCover</td>
</tr>
<tr>
<td><code>zOffset</code></td>
<td>GroundCover</td>
</tr>
</tbody>
</table>
## GroundPlane Member List

This is the complete list of members for `GroundPlane`, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code></td>
<td><code>SimObject fromObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call</code></td>
<td><code>string method, string args...</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clearScopeToClient</code></td>
<td><code>NetConnection client</code></td>
</tr>
<tr>
<td><code>clone</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump</code></td>
<td><code>bool detailed=false</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCanSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClientObject</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField</code></td>
<td><code>int index</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getEulerRotation</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method/Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>GroundPlane [static]</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>GroundPlane [static]</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>Material</td>
<td>GroundPlane</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>postApply()</td>
<td>GroundPlane</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>scaleU</td>
<td>GroundPlane</td>
</tr>
<tr>
<td>scaleV</td>
<td>GroundPlane</td>
</tr>
<tr>
<td>schedule(time, method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(fieldName, type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(fieldName, value, index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>squareSize</code></td>
<td><code>GroundPlane</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td><code>SceneObject</code></td>
</tr>
</tbody>
</table>
GuiAutoScrollCtrl Member List

This is the complete list of members for GuiAutoScrollCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Member Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>childBorder</td>
<td>GuiAutoScrollCtrl</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class Name</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isLooping</td>
<td>GuiAutoScrollCtrl</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onComplete()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>---------------------------------------</td>
</tr>
<tr>
<td><code>onDialogPush()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onGainFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onLoseFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onObjectAdded(SimObject object)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onObjectRemoved(SimObject object)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onRemove()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onReset()</code></td>
<td>GuiAutoScrollCtrl</td>
</tr>
<tr>
<td><code>onSleep()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onStart()</code></td>
<td>GuiAutoScrollCtrl</td>
</tr>
<tr>
<td><code>onTick()</code></td>
<td>GuiAutoScrollCtrl</td>
</tr>
<tr>
<td><code>onVisible(bool state)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onWake()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>pointInControl(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>position</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>profile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>pushToBack(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>remove(SimObject objects...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>reorderChild(SimObject child1, SimObject child2)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>reset()</code></td>
<td>GuiAutoScrollCtrl</td>
</tr>
<tr>
<td><code>resetDelay</code></td>
<td>GuiAutoScrollCtrl</td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>scrollDirection</code></td>
<td>GuiAutoScrollCtrl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>scrollOutOfSight</td>
<td>GuiAutoScrollCtrl</td>
</tr>
<tr>
<td>scrollSpeed</td>
<td>GuiAutoScrollCtrl</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProcessTicks(bool tick) void</td>
<td>GuiTickCtrl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>setValue(string value)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setVisible(bool state=true)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>sort(string callbackFunction)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>startDelay</code></td>
<td>GuiAutoScrollCtrl</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>tooltip</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>tooltipProfile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>variable</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>vertSizing</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>visible</code></td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
GuiBitmapBorderCtrl Member List

This is the complete list of members for `GuiBitmapBorderCtrl`, including all inherited members.

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecursive(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsIsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>SimObject</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getld()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td><code>isMember(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMouseLocked()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isVisible()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>langTableMod</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>listObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>makeFirstResponder(bool isFirst)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>minExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>modal</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>onAction()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onActive(bool state)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onAdd()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragEnter(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragExit(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragged(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDropped(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onDialogPop()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onDialogPush()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onGainFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onLoseFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>onObjectAdded</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onObjectRemoved</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onRemove()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onSleep()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onVisible(bool state)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onWake()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>pointInControl(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>position</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>profile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>pushToBack(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>remove(SimObject objects...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>reorderChild(SimObject child1, SimObject child2)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setClassNameSpace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
GuiBitmapButtonCtrl Member List

This is the complete list of members for GuiBitmapButtonCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>autoFitExtents</td>
<td>GuiBitmapButtonCtrl</td>
</tr>
<tr>
<td>bitmap</td>
<td>GuiBitmapButtonCtrl</td>
</tr>
<tr>
<td>bitmapMode</td>
<td>GuiBitmapButtonCtrl</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>buttonType</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Command</td>
<td>Type</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td><code>command</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>controlsChild(GuiControl control)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>extent</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findHitControl(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findHitControls(int x, int y, int width, int height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findObjectByInternalName(string internalName, bool searchChildren=false)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getAspect()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td><code>getFieldType</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getFullCount</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getGlobalCenter</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGlobalPosition</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getMinExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getParent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getPosition</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getRandom</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getRoot</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getText()</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>getValue</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>groupNum</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isActive</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Function Call</td>
<td>Type</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAltClick()</td>
<td>GuiBitmapButtonCtrl</td>
</tr>
<tr>
<td>onClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>onControlDragExit(GuiControl control, Point2I dropPoint)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onControlDragged(GuiControl control, Point2I dropPoint)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onControlDropped(GuiControl control, Point2I dropPoint)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onCtrlClick()</code></td>
<td>GuiBitmapButtonCtrl</td>
</tr>
<tr>
<td><code>onDefaultClick()</code></td>
<td>GuiBitmapButtonCtrl</td>
</tr>
<tr>
<td><code>onDialogPop()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onDialogPush()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onDoubleClick()</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>onGainFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onLoseFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onMouseDown()</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>onMouseDragged()</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>onMouseEnter()</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>onMouseLeave()</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>onMouseUp()</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>onObjectAdded(SimObject object)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onObjectRemoved(SimObject object)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onRemove()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onRightClick()</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>onShiftClick()</code></td>
<td>GuiBitmapButtonCtrl</td>
</tr>
<tr>
<td><code>onSleep()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onVisible(bool state)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onWake()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>performClick()</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>pointInControl(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>position</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>profile</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>pushToBack(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>remove(SimObject objects...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>reorderChild(SimObject child1, SimObject child2)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>resetState()</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setBitmap(string path)</code></td>
<td><code>GuiBitmapButtonCtrl</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setStateOn(bool isOn=true)</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setText(string text)</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>setTextID(string id)</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>text</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>textID</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>useModifiers</td>
<td>GuiBitmapButtonCtrl</td>
</tr>
<tr>
<td>useMouseEvents</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>useStates</td>
<td>GuiBitmapButtonCtrl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
**GuiBitmapButtonTextCtrl Member List**

This is the complete list of members for `GuiBitmapButtonTextCtrl`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>autoFitExtents</td>
<td>GuiBitmapButtonCtrl</td>
</tr>
<tr>
<td>bitmap</td>
<td>GuiBitmapButtonCtrl</td>
</tr>
<tr>
<td>bitmapMode</td>
<td>GuiBitmapButtonCtrl</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>buttonType</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Command</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td><code>command</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>controlsChild(GuiControl control)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>extent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControl(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControls(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findObjectByInternalName(string internalName, bool searchChildren=false)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getAspect()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function Name</td>
<td>Type</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>getFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>getText</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>groupNum</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAltClick()</td>
<td>GuiBitmapButtonCtrl</td>
</tr>
<tr>
<td>onClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td><code>onControlDragExit</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onControlDragged</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onControlDropped</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onCtrlClick</code></td>
<td>GuiBitmapButtonCtrl</td>
</tr>
<tr>
<td><code>onDefaultClick</code></td>
<td>GuiBitmapButtonCtrl</td>
</tr>
<tr>
<td><code>onDialogPop</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onDialogPush</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onDoubleClick</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>onGainFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onLoseFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onMouseDown</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>onMouseDragged</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>onMouseEnter</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>onMouseLeave</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>onMouseUp</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>onObjectAdded</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onObjectRemoved</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onRemove</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onRightClick</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>onShiftClick</code></td>
<td>GuiBitmapButtonCtrl</td>
</tr>
<tr>
<td><code>onSleep</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onVisible</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onWake</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>performClick</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>pointInControl</code>(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>position</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>profile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>pushToBack</code> (SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>remove</code> (SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>reorderChild</code> (SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>resetState</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>resize</code> (int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>save</code> (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule</code> (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setActive</code> (bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setBitmap</code> (string path)</td>
<td>GuiBitmapButtonCtrl</td>
</tr>
<tr>
<td><code>setCanSave</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCenter</code> (int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setExtent</code> (Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setExtent</code> (S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFirstResponder</code> ()</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code> setHidden</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setPosition</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setPositionGlobal</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setProfile</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setStateOn</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setText</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>setTextID</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>setValue</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setVisible</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>sort</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>text</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>textID</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>tooltip</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>tooltipProfile</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>useModifiers</code></td>
<td><code>GuiBitmapButtonCtrl</code></td>
</tr>
<tr>
<td><code>useMouseEvents</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>useStates</code></td>
<td><code>GuiBitmapButtonCtrl</code></td>
</tr>
<tr>
<td><code>variable</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>vertSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>visible</code></td>
<td><code>GuiControl</code></td>
</tr>
</tbody>
</table>
GuiBitmapCtrl Member List

This is the complete list of members for GuiBitmapCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bitmap</td>
<td>GuiBitmapCtrl</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
</tbody>
</table>
delete()                SimObject
deleteAllObjects()     SimSet
dump(bool detailed=false)     SimObject
dumpClassHierarchy()    SimObject
dumpGroupHierarchy()    SimObject
dumpMethods()          SimObject
extent                  GuiControl
findHitControl(int x, int y)        GuiControl
findHitControls(int x, int y, int width, int height)        GuiControl
findObjectByInternalName(string internalName, bool searchChildren=false) SimSet
getAspect()            GuiControl
getCanSave()           SimObject
getCenter()            GuiControl
getClassName()         SimObject
getClassNamespace()    SimObject
count()                SimSet
getDebugInfo()         SimObject
getDeclarationLine()   SimObject
getDynamicField(int index)    SimObject
getDynamicFieldCount()  SimObject
getExtent()            GuiControl
getField(int index)    SimObject
getFieldCount()        SimObject
getFieldType(string fieldName)    SimObject
getFieldValue(string fieldName, int index=-1) SimObject
getFilename()          SimObject
getFirstResponder()     GuiControl
<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>getFullCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getGlobalCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGlobalPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getMinExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getParent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getRoot()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getValue()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isActive</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isContainer</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setBitmap(String filename)</td>
<td>GuiBitmapCtrl</td>
</tr>
<tr>
<td>setBitmap(String filename, bool resize)</td>
<td>GuiBitmapCtrl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code> setHidden</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setPosition</code> (int x, int y)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setPositionGlobal</code> (int x, int y)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setProfile</code> (GuiControlProfile profile)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setValue</code> (int x, int y)</td>
<td><code>GuiBitmapCtrl</code></td>
</tr>
<tr>
<td><code>GuiControl::setValue</code> (string value)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setVisible</code> (bool state=true)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>sort</code> (string callbackFunction)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>tooltip</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>tooltipProfile</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>variable</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>vertSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>visible</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>wrap</code></td>
<td><code>GuiBitmapCtrl</code></td>
</tr>
</tbody>
</table>
Copyright © GarageGames, LLC. All Rights Reserved.
GuiBorderButtonCtrl Member List

This is the complete list of members for GuiBorderButtonCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>buttonType</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsIsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>objectId(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getText()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>getValue()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>groupNum</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDoubleClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onMouseDown()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseDragged()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseEnter()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseLeave()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseUp()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onRightClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>performClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resetState()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>Method</td>
<td>Category</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setStateOn(bool isOn=true)</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setText</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>setTextID</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>setValue</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setVisible</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>sort</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>text</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>textID</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>tooltip</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>tooltipProfile</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>useMouseEvents</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>variable</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>vertSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>visible</code></td>
<td><code>GuiControl</code></td>
</tr>
</tbody>
</table>
GuiBubbleTextCtrl Member List

This is the complete list of members for GuiBubbleTextCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>anchorBottom</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorLeft</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorRight</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorTop</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>docking</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getGlobalCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGlobalPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getMinExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getParent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getRoot()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getValue()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isActive()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isContainer</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>maxLength</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>onControlDragged</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>padding</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>SimSet</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setText(string text)</code></td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td><code>setTextID(string textID)</code></td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td><code>setValue(string value)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setVisible(bool state=true)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>sort(string callbackFunction)</code></td>
<td>SimSet</td>
</tr>
</tbody>
</table>

`superClass` SimObject
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>textID</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
GuiButtonBaseCtrl Member List

This is the complete list of members for GuiButtonBaseCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>buttonType</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getText()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>getValue()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>groupNum</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>isField</code> (string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMember</code> (SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>isMemberOfClass</code> (string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod</code> (string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMouseLocked</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isVisible</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>langTableMod</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>listObjects</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>makeFirstResponder</code> (bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>minExtent</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>modal</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>onAction</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onActive</code> (bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onAdd</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onClick</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>onControlDragEnter</code> (GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onControlDragExit</code> (GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onControlDragged</code> (GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onControlDropped</code> (GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDoubleClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onMouseDown()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseDragged()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseEnter()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseLeave()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseUp()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onRightClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>performClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resetState()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setStateOn(bool isOn=true)</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setText</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>setTextID</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>setValue</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setVisible</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>sort</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>text</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>textID</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>tooltip</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>tooltipProfile</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>useMouseEvents</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>variable</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>vertSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>visible</code></td>
<td><code>GuiControl</code></td>
</tr>
</tbody>
</table>
GuiButtonCtrl Member List

This is the complete list of members for GuiButtonCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>buttonType</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getText()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>groupNum</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method/Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>isField</code> (string fieldName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isFirstResponder</code> ()</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMember</code> (SimObject obj)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>isMemberOfClass</code> (string className)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod</code> (string methodName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMouseLocked</code> ()</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isVisible</code> ()</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>langTableMod</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>listObjects</code> ()</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>makeFirstResponder</code> (bool isFirst)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>minExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>modal</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>onAction</code> ()</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onActive</code> (bool state)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onAdd</code> ()</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onClick</code> ()</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragEnter</code> (GuiControl control, Point2I dropPoint)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragExit</code> (GuiControl control, Point2I dropPoint)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragged</code> (GuiControl control, Point2I dropPoint)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDropped</code> (GuiControl control, Point2I dropPoint)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Function Name</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDoubleClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onMouseDown()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseDragged()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseEnter()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseLeave()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseUp()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onObjectAdded()</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved()</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onRightClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>performClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resetState()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code> setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setStateOn(bool isOn=true)</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setText</code> (string text)</td>
<td>GuiButtonItemCtrl</td>
</tr>
<tr>
<td><code>setTextID</code> (string id)</td>
<td>GuiButtonItemCtrl</td>
</tr>
<tr>
<td><code>setValue</code> (string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setVisible</code> (bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>sort</code> (string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>text</td>
<td>GuiButtonItemCtrl</td>
</tr>
<tr>
<td>textID</td>
<td>GuiButtonItemCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>useMouseEvents</td>
<td>GuiButtonItemCtrl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## GuiCanvas Member List

This is the complete list of members for **GuiCanvas**, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>alwaysHandleMouseButtons</td>
<td>GuiCanvas</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecursive(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clientToScreen(Point2I coordinate)</td>
<td>GuiCanvas</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>cursorOff()</td>
<td>GuiCanvas</td>
</tr>
<tr>
<td>cursorOn()</td>
<td>GuiCanvas</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClass()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getContent()</td>
<td>GuiCanvas</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getCursorPos()</td>
<td>GuiCanvas</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiCanvas</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
</tbody>
</table>
getFieldCount
getFieldType(string fieldName)
getFieldValue(string fieldName, int index=-1)
getFilename()
getFirstResponder()
getFullCount()
getGlobalCenter()
getGlobalPosition()
getGroup()
getld()
getInternalName()
getMinExtent()
getMode(int modeId)
getModeCount()
getMouseControl()
getName()
getObject(int index)
getObjectIndex(SimObject obj)
getParent()
getPosition()
getRandom()
getRoot()
getSuperClassNamespace()
getValue
getVideoMode()
getWindowPosition()
hidden
hideCursor()
<table>
<thead>
<tr>
<th>Function</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isCursorOn()</td>
<td>GuiCanvas</td>
</tr>
<tr>
<td>isCursorShown()</td>
<td>GuiCanvas</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isFullscreen()</td>
<td>GuiCanvas</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMaximized()</td>
<td>GuiCanvas</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMinimized()</td>
<td>GuiCanvas</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>maximizeWindow()</td>
<td>GuiCanvas</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minimizeWindow()</td>
<td>GuiCanvas</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>numFences</td>
<td>GuiCanvas</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Module</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td><code>pointInControl</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>popDialog()</code></td>
<td><code>GuiCanvas</code></td>
</tr>
<tr>
<td><code>popDialog(GuiControl ctrl)</code></td>
<td><code>GuiCanvas</code></td>
</tr>
<tr>
<td><code>popLayer(S32 layer)</code></td>
<td><code>GuiCanvas</code></td>
</tr>
<tr>
<td><code>popLayer()</code></td>
<td><code>GuiCanvas</code></td>
</tr>
<tr>
<td><code>position</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>profile</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>pushDialog(GuiControl ctrl, int layer=0, bool center=false)</code></td>
<td><code>GuiCanvas</code></td>
</tr>
<tr>
<td><code>pushToBack(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>remove(SimObject objects...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>renderFront(bool enable)</code></td>
<td><code>GuiCanvas</code></td>
</tr>
<tr>
<td><code>reorderChild(SimObject child1, SimObject child2)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>repaint(int elapsedMS=0)</code></td>
<td><code>GuiCanvas</code></td>
</tr>
<tr>
<td><code>reset()</code></td>
<td><code>GuiCanvas</code></td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>restoreWindow()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>screenToClient(Point2I coordinate)</code></td>
<td><code>GuiCanvas</code></td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setContent(GuiControl ctrl)</code></td>
<td><code>GuiCanvas</code></td>
</tr>
<tr>
<td><code>setCursorPosition(GuiCursor cursor)</code></td>
<td><code>GuiCanvas</code></td>
</tr>
<tr>
<td><code>setCursorPositionPos(F32 posX, F32 posY)</code></td>
<td><code>GuiCanvas</code></td>
</tr>
</tbody>
</table>
setCursorPos(Point2I pos)
setEditorOnly(bool value=true)
setExtent(Point2I p)
setExtent(S32 width, S32 height)
setFieldType(string fieldName, string type)
setFieldValue(string fieldName, string value, int index=-1)
setFilename(string fileName)
setFirstResponder
setFirstResponder()
setFocus()
setHidden(bool value=true)
setInternalName(string newInternalName)
setIsExpanded(bool state=true)
setIsSelected(bool state=true)
setLocked(bool value=true)
setName(string newName)
setNameChangeAllowed(bool value=true)
setPosition(int x, int y)
setPositionGlobal(int x, int y)
setProfile(GuiControlProfile profile)
setSuperClassNamespace(string name)
setValue(string value)
setVideoMode(int width, int height, bool fullscreen, [int bitDepth], [int refreshRate], [int antialiasLevel])
setVisible(bool state=true)
setWindowPosition(Point2I position)
setWindowTitle(string newTitle)
<table>
<thead>
<tr>
<th>Function</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>showCursor()</code></td>
<td>GuiCanvas</td>
</tr>
<tr>
<td><code>sort(callbackFunction)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>toggleFullscreen()</code></td>
<td>GuiCanvas</td>
</tr>
<tr>
<td><code>tooltip</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>tooltipProfile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>variable</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>vertSizing</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>visible</code></td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
GuiCheckBoxCtrl Member List

This is the complete list of members for GuiCheckBoxCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>buttonType</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsIsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
getFullCount() SimSet
getGlobalCenter() GuiControl
getGlobalPosition() GuiControl
getGroup() SimObject
getId() SimObject
getInternalName() SimObject
getMinExtent() GuiControl
getName() SimObject
getObject(int index) SimSet
getObjectIndex(SimObject obj) SimSet
getParent() GuiControl
getPosition() GuiControl
getRandom() SimSet
getRoot() GuiControl
getSuperClassNamespace() SimObject
text() GuiButtonBaseCtrl
getValue GuiControl
groupNum GuiButtonBaseCtrl
hidden SimObject
horizSizing GuiControl
hovertime GuiControl
internalName SimObject
isActive GuiControl
isAwake() GuiControl
isChildOfGroup(SimGroup group) SimObject
isContainer GuiControl
isEditorOnly() SimObject
isExpanded() SimObject
<table>
<thead>
<tr>
<th>Function</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMember(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMouseLocked()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isStateOn()</code></td>
<td><code>GuiCheckBoxCtrl</code></td>
</tr>
<tr>
<td><code>isVisible()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>langTableMod</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>listObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>makeFirstResponder(bool isFirst)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>minExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>modal</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>onAction()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onActive(bool state)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onAdd()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onClick()</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>onControlDragEnter(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragExit(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragged(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDropped(GuiControl control, Point2I</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>dropPoint()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDoubleClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onMouseDown()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseDragged()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseEnter()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseLeave()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseUp()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onRightClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>performClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td><code>resetState()</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>setStateOn(bool newState)</td>
<td>GuiCheckBoxCtrl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setText(string text)</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>setTextID(string id)</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>text</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>textID</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>useMouseEvents</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
# GuiChunkedBitmapCtrl Member List

This is the complete list of members for `GuiChunkedBitmapCtrl`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>accelerator</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>acceptsAsChild(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>active</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>add(SimObject objects...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>addGuiControl(GuiControl control)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>altCommand</code></td>
<td><code> GuiControl</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>bitmap</code></td>
<td><code>GuiChunkedBitmapCtrl</code></td>
</tr>
<tr>
<td><code>bringToFront(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>callOnChildren(string method, string args...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>callOnChildrenNoRecurse(string method, string args...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clear()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>clearFirstResponder(bool ignored=false)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>command</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>controlsChild(GuiControl control)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
onDialogPush() GuiControl
onGainFirstResponder() GuiControl
onLoseFirstResponder() GuiControl
onObjectAdded(SimObject object) SimSet
onObjectRemoved(SimObject object) SimSet
onRemove() GuiControl
onSleep() GuiControl
onVisible(bool state) GuiControl
onWake() GuiControl
parentGroup SimObject
persistentId SimObject
pointInControl(int x, int y) GuiControl
position GuiControl
profile GuiControl
pushToBack(SimObject obj) SimSet
remove(SimObject objects...) SimSet
reorderChild(SimObject child1, SimObject child2) SimSet
resize(int x, int y, int width, int height) GuiControl
save(string fileName, bool selectedOnly=false, string preAppendString="") SimObject
schedule(float time, string method, string args...) SimObject
setActive(bool state=true) GuiControl
setBitmap(string filename) GuiChunkedBitmapCtrl
setCanSave(bool value=true) SimObject
setCenter(int x, int y) GuiControl
<table>
<thead>
<tr>
<th>Method</th>
<th>Class/Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>setClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tile</td>
<td>GuiChunkedBitmapCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>useVariable</td>
<td>GuiChunkedBitmapCtrl</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# GuiClockHud Member List

This is the complete list of members for `GuiClockHud`, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsIsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>SimObject</td>
</tr>
<tr>
<td>fillColor</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>frameColor</td>
<td>GuiClockHud</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTime()</td>
<td>GuiClockHud</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td><code>isField</code> (string fieldName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isFirstResponder</code> ()</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMember</code> (<code>SimObject</code> obj)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>isMemberOfClass</code> (string className)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod</code> (string methodName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMouseLocked</code> ()</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isVisible</code> ()</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>langTableMod</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>listObjects</code> ()</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>makeFirstResponder</code> (bool isFirst)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>minExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>modal</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>onAction</code> ()</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onActive</code> (bool state)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onAdd</code> ()</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragEnter</code> (GuiControl control, Point2I dropPoint)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragExit</code> (GuiControl control, Point2I dropPoint)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragged</code> (GuiControl control, Point2I dropPoint)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDropped</code> (GuiControl control, Point2I dropPoint)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onDialogPop</code> ()</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNameSpace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setReverseTime(float timeInSeconds=60)</td>
<td>GuiClockHud</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTime(float timeInSeconds=60)</td>
<td>GuiClockHud</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>showFill</td>
<td>GuiClockHud</td>
</tr>
<tr>
<td>showFrame</td>
<td>GuiClockHud</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>textColor</td>
<td>GuiClockHud</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>--------------</td>
<td>------------</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
GuiConsole Member List

This is the complete list of members for GuiConsole, including all inherited members.

```cpp
onMessageSelected(ConsoleLogEntry::Level level, string message)
```

Copyright © GarageGames, LLC. All Rights Reserved.
## GuiConsoleEditCtrl Member List

This is the complete list of members for *GuiConsoleEditCtrl*, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>anchorBottom</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorLeft</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorRight</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorTop</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecursive(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clearSelectedText()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>deniedSound</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>docking</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>escapeCommand</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>forceValidateText()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getCursorPos()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getText()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>historySize</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAllTextSelected()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>maxLength</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>-----------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onReturn()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onTabComplete(string val)</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>onValidate()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>padding</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>password</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>passwordMask</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>selectAllText()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCursorPos(int position)</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setText(string text)</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>setTextID(string textID)</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sinkAllKeyEvents</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tabComplete</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>text</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>textID</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>useSiblingScroller</td>
<td>GuiConsoleEditCtrl</td>
</tr>
<tr>
<td>validate</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
## GuiContainer Member List

This is the complete list of members for *GuiContainer*, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>anchorBottom</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorLeft</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorRight</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorTop</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>command</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>controlsChild(GuiControl control)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>docking</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>extent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControl(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControls(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findObjectByInternalName(string internalName, bool searchChildren=false)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getAspect()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getIndex()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>getFieldType</code></td>
<td><code>getFieldValue</code></td>
</tr>
<tr>
<td><code>getFieldValue</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td><code>getFirstResponder</code></td>
</tr>
<tr>
<td><code>getFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getFullCount</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getGlobalCenter</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGlobalPosition</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getMinExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObject</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getObjectIndex</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getParent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getPosition</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getRandom</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getRoot</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getValue</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isActive</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isAwake</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isChildOfGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isContainer</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimSet</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>onControlDropped</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>padding</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setExtent</code> (Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setExtent</code> (S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setPosition</code> (int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setPositionGlobal</code> (int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setProfile</code> (GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setValue</code> (string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setVisible</code> (bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>sort</code> (string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>tooltip</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>tooltipProfile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>variable</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>------------</td>
<td>------------</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# GuiControl Member List

This is the complete list of members for GuiControl, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsIsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>extent</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findHitControl(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findHitControls(int x, int y, int width, int height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findObjectByInternalName(string internalName, bool searchChildren=false)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getAspect()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class Name</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td><code>onObjectAdded</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>onObjectRemoved</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>onRemove</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onSleep</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onVisible</code> (bool state)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onWake</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>pointInControl</code> (int x, int y)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>position</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>profile</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>pushToBack</code> (SimObject obj)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>remove</code> (SimObject objects...)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>reorderChild</code> (SimObject child1, SimObject child2)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>resize</code> (int x, int y, int width, int height)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>save</code> (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule</code> (float time, string method, string args...)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setActive</code> (bool state=true)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setCanSave</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter</code> (int x, int y)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setExtent</code> (Point2I p)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent</code> (S32 width, S32 height)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassName(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
## GuiControlArrayControl Member List

This is the complete list of members for `GuiControlArrayControl`, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>colCount</td>
<td>GuiControlArrayControl</td>
</tr>
<tr>
<td>colSizes</td>
<td>GuiControlArrayControl</td>
</tr>
<tr>
<td>colSpacing</td>
<td>GuiControlArrayControl</td>
</tr>
<tr>
<td>command</td>
<td>type</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>command</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>controllsChild(GuiControl control)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>extent</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findHitControl(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findHitControls(int x, int y, int width, int height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findObjectByInternalName(string internalName, bool searchChildren=false)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getAspect()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>getFieldType</code> (string fieldName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue</code> (string fieldName, int index=-1)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getFullCount</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getGlobalCenter</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGlobalPosition</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getMinExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObject</code> (int index)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getObjectIndex</code> (SimObject obj)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getParent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getPosition</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getRandom</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getRoot</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getValue</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isActive</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isChildOfGroup</code> (SimGroup group)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>onControlDropped</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>rowSize</td>
<td>GuiControlArrayControl</td>
</tr>
<tr>
<td>rowSpacing</td>
<td>GuiControlArrayControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setValue(string value)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setVisible(bool state=true)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>sort(string callbackFunction)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# GuiControlProfile Member List

This is the complete list of members for `GuiControlProfile`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>autoSizeHeight</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>autoSizeWidth</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>bevelColorHL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>bevelColorLL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>bitmap</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>border</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>borderColor</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>borderColorHL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>borderColorNA</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>borderThickness</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>call</td>
<td>SimObject</td>
</tr>
<tr>
<td>canKeyFocus</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>category</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>cursorColor</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>fillColor</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fillColorHL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fillColorNA</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fillColorSEL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontCharset</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontColor</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontColorHL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontColorLink</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontColorLinkHL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontColorLinkNA</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontColors</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontColorSEL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontSize</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontType</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getStringWidth(pString)</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hasBitmapArray</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>justify</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>mouseOverSelected</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>numbersOnly</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>opaque</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class Name</td>
</tr>
<tr>
<td>---------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>profileForChildren</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>returnTab</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>save(fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(time, method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNameNamespace(name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(fieldName, type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>soundButtonDown</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>soundButtonOver</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tab</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>textOffset</td>
<td>GuiControlProfile</td>
</tr>
</tbody>
</table>
## GuiCrossHairHud Member List

This is the complete list of members for **GuiCrossHairHud**, including all inherited members.

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Class Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bitmap</td>
<td>GuiBitmapCtrl</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsIsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>damageFillColor</td>
<td>GuiCrossHairHud</td>
</tr>
<tr>
<td>damageFrameColor</td>
<td>GuiCrossHairHud</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>damageOffset</td>
<td>GuiCrossHairHud</td>
</tr>
<tr>
<td>damageRect</td>
<td>GuiCrossHairHud</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getFieldType</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getFullCount</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getGlobalCenter</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGlobalPosition</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getMinExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObject</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getObjectIndex</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getParent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getPosition</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getRandom</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getRoot</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getValue</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isActive</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isAwake</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isChildOfGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isContainer</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Method Name</td>
<td>Class Name</td>
</tr>
<tr>
<td>------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>dropPoint()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setBitmap(String filename)</td>
<td>GuiBitmapCtrl</td>
</tr>
<tr>
<td>setBitmap(String filename, bool resize)</td>
<td>GuiBitmapCtrl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldTexture(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(int x, int y)</td>
<td>GuiBitmapCtrl</td>
</tr>
<tr>
<td>GuiControl::setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>Variable</td>
<td>Class</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
<tr>
<td>wrap</td>
<td>GuiBitmapCtrl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
GuiCursor Member List

This is the complete list of members for GuiCursor, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bitmapName</td>
<td>GuiCursor</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td><code>getFieldType</code> (string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue</code> (string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hotSpot</code></td>
<td>GuiCursor</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup</code> (SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField</code> (string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass</code> (string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod</code> (string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>renderOffset</code></td>
<td>GuiCursor</td>
</tr>
<tr>
<td><code>save</code> (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule</code> (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Parameters</td>
</tr>
<tr>
<td>-------------</td>
<td>------------</td>
</tr>
<tr>
<td>setCanSave</td>
<td><code>bool</code> value=true</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td><code>string</code> name</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td><code>bool</code> value=true</td>
</tr>
<tr>
<td>setFieldType</td>
<td><code>string</code> fieldName, <code>string</code> type</td>
</tr>
<tr>
<td>setFieldValue</td>
<td><code>string</code> fieldName, <code>string</code> value, <code>int</code> index=-1</td>
</tr>
<tr>
<td>setFilename</td>
<td><code>string</code> fileMane</td>
</tr>
<tr>
<td>setHidden</td>
<td><code>bool</code> value=true</td>
</tr>
<tr>
<td>setInternalName</td>
<td><code>string</code> newInternalName</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td><code>bool</code> state=true</td>
</tr>
<tr>
<td>setIsSelected</td>
<td><code>bool</code> state=true</td>
</tr>
<tr>
<td>setLocked</td>
<td><code>bool</code> value=true</td>
</tr>
<tr>
<td>setName</td>
<td><code>string</code> newName</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td><code>bool</code> value=true</td>
</tr>
<tr>
<td>setSuperClassNamespace</td>
<td><code>string</code> name</td>
</tr>
<tr>
<td>superClass</td>
<td></td>
</tr>
</tbody>
</table>
## GuiDirectoryFileListCtrl Member List

This is the complete list of members for `GuiDirectoryFileListCtrl`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addFilteredItem(string newItem)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>allowMultipleSelections</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clearItemColor(int index)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>clearItems()</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>clearSelection()</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>colorBullet</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>deleteItem(int itemIndex)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>doMirror()</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>fileFilter</td>
<td>GuiDirectoryFileListCtrl</td>
</tr>
<tr>
<td>filePath</td>
<td>GuiDirectoryFileListCtrl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findItemText(string findText, bool bCaseSensitive=false)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>fitParentWidth</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getItemCount()</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>getItemCount(int index)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>getItemText(int index)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>getLastClickItem()</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSelCount()</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>getSelectedFile()</td>
<td>GuiDirectoryFileListCtrl</td>
</tr>
<tr>
<td>getSelectedFiles()</td>
<td>GuiDirectoryFileListCtrl</td>
</tr>
<tr>
<td>getSelectedItem()</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>getSelectedItems()</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>insertItem(string text, int index)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>isMethod</code> (string methodName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMouseLocked()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isObjectMirrored</code> (string indexIdString)</td>
<td><code>GuiListBoxCtrl</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isVisible()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>langTableMod</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>listObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>makeFirstResponder</code> (bool isFirst)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>makeNameCallback</code></td>
<td><code>GuiListBoxCtrl</code></td>
</tr>
<tr>
<td><code>minExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>mirrorSet</code></td>
<td><code>GuiListBoxCtrl</code></td>
</tr>
<tr>
<td><code>modal</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>onAction()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onActive</code> (bool state)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onAdd()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onClearSelection()</code></td>
<td><code>GuiListBoxCtrl</code></td>
</tr>
<tr>
<td><code>onControlDragEnter</code> (GuiControl control, Point2I dropPoint)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragExit</code> (GuiControl control, Point2I dropPoint)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragged</code> (GuiControl control, Point2I dropPoint)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDropped</code> (GuiControl control, Point2I dropPoint)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onDeleteKey()</code></td>
<td><code>GuiListBoxCtrl</code></td>
</tr>
<tr>
<td><code>onDialogPop()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Method</td>
<td>Object Type</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDoubleClick()</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onMouseDragged()</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>onMouseUp(string itemHit, string mouseClickCount)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSelect(string index, string itemText)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onUnselect(string index, string itemText)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reload()</td>
<td>GuiDirectoryFileListCtrl</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>removeFilteredItem(string itemName)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string)</td>
<td>SimObject</td>
</tr>
<tr>
<td>preAppendString=&quot;&quot;</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td></td>
</tr>
<tr>
<td><strong>schedule</strong> (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setActive</strong> (bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>setCanSave</strong> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setCenter</strong> (int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>setClassNamespace</strong> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setCurSel</strong> (int indexId)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td><strong>setCurSelRange</strong> (int indexStart, int indexStop=999999)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td><strong>setEditorOnly</strong> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setExtent</strong> (Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>setExtent</strong> (S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>setFieldType</strong> (string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setFieldValue</strong> (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setFilename</strong> (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setFilter</strong> (string filter)</td>
<td>GuiDirectoryFileListCtrl</td>
</tr>
<tr>
<td><strong>setFirstResponder</strong></td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>setFirstResponder</strong> ()</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>setHidden</strong> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setInternalName</strong> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setIsExpanded</strong> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setIsSelected</strong> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setItemColor</strong> (int index, ColorF color)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td><strong>setItemText</strong> (int index, string newText)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td><strong>setItemTooltip</strong> (int index, string text)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td><strong>setLocked</strong> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>setMultipleSelection(bool allowMultSelections)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPath(string path, string filter)</td>
<td>GuiDirectoryFileListCtrl</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSelected(int index, bool setSelected=true)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>setSuperClassNameamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
## GuiDragAndDropControl Member List

This is the complete list of members for `GuiDragAndDropControl`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecursive(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsIsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>deleteOnMouseUp</td>
<td>GuiDragAndDropControl</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control,</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><code>Point2I dropPoint()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onDialogPop()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onDialogPush()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onGainFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onLoseFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onObjectAdded(SimObject object)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>onObjectRemoved(SimObject object)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>onRemove(SimObject object)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onSleep()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onVisible(bool state)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onWake()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>pointInControl(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>position</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>profile</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>pushToBack(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>remove(SimObject objects...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>reorderChild(SimObject child1, SimObject child2)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Method Name</td>
<td>Class/Function</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setExtent</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setExtent</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setPosition</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setPositionGlobal</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setProfile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setValue</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setVisible</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>sort</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>startDragging</code></td>
<td>GuiDragAndDropControl</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td></td>
<td>GuiControl</td>
</tr>
<tr>
<td>------</td>
<td>------------</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## GuiDynamicCtrlArrayControl Member List

This is the complete list of members for `GuiDynamicCtrlArrayControl`, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td>active</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>altCommand</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>autoCellSize</td>
<td><code>GuiDynamicCtrlArrayControl</code></td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td>canSave</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>class</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>className</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>clear()</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>clone()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>colCount</td>
<td><code>GuiDynamicCtrlArrayControl</code></td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>colSize</td>
<td>GuiDynamicCtrlArrayControl</td>
</tr>
<tr>
<td>colSpacing</td>
<td>GuiDynamicCtrlArrayControl</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dynamicSize</td>
<td>GuiDynamicCtrlArrayControl</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>fillRowFirst</td>
<td>GuiDynamicCtrlArrayControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>frozen</td>
<td>GuiDynamicCtrlArrayControl</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getGlobalCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGlobalPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getMinExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getParent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getRoot()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getValue</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>onAdd()</td>
<td>UIControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>padding</td>
<td>GuiDynamicCtrlArrayControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>refresh()</td>
<td>GuiDynamicCtrlArrayControl</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>reorderChild</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize</td>
<td>GuiControl</td>
</tr>
<tr>
<td>rowCount</td>
<td>GuiDynamicCtrlArrayControl</td>
</tr>
<tr>
<td>rowSize</td>
<td>GuiDynamicCtrlArrayControl</td>
</tr>
<tr>
<td>rowSpacing</td>
<td>GuiDynamicCtrlArrayControl</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setValue(string value)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setVisible(bool state=true)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>sort(string callbackFunction)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>tooltip</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>tooltipProfile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>variable</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>vertSizing</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>visible</code></td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
GuiFadeinBitmapCtrl Member List

This is the complete list of members for GuiFadeinBitmapCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method/Function</th>
<th>Class/Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bitmap</td>
<td>GuiBitmapCtrl</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>click()</td>
<td>GuiFadeinBitmapCtrl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>done</td>
<td>GuiFadeinBitmapCtrl</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>fadeColor</td>
<td>GuiFadeinBitmapCtrl</td>
</tr>
<tr>
<td>fadeInEase</td>
<td>GuiFadeinBitmapCtrl</td>
</tr>
<tr>
<td>fadeInTime</td>
<td>GuiFadeinBitmapCtrl</td>
</tr>
<tr>
<td>fadeOutEase</td>
<td>GuiFadeinBitmapCtrl</td>
</tr>
<tr>
<td>fadeOutTime</td>
<td>GuiFadeinBitmapCtrl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>onControlDragExit</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDone()</td>
<td>GuiFadeinBitmapCtrl</td>
</tr>
<tr>
<td>onGainFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded()</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved()</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setBitmap(String filename)</code></td>
<td><code>GuiBitmapCtrl</code></td>
</tr>
<tr>
<td><code>setBitmap(String filename, bool resize)</code></td>
<td><code>GuiBitmapCtrl</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>setValue(int x, int y)</td>
<td>GuiBitmapCtrl</td>
</tr>
<tr>
<td>GuiControl::setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
<tr>
<td>waitTime</td>
<td>GuiFadeinBitmapCtrl</td>
</tr>
<tr>
<td>wrap</td>
<td>GuiBitmapCtrl</td>
</tr>
</tbody>
</table>
## GuiFrameSetCtrl Member List

This is the complete list of members for `GuiFrameSetCtrl`, including all inherited members.

<table>
<thead>
<tr>
<th>Method/Property</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>accelerator</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>acceptsAsChild(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>active</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>add(SimObject objects...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>addColumn()</code></td>
<td><code>GuiFrameSetCtrl</code></td>
</tr>
<tr>
<td><code>addGuiControl(GuiControl control)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>addRow()</code></td>
<td><code>GuiFrameSetCtrl</code></td>
</tr>
<tr>
<td><code>altCommand</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>anchorBottom</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>anchorLeft</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>anchorRight</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>anchorTop</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>autoBalance</code></td>
<td><code>GuiFrameSetCtrl</code></td>
</tr>
<tr>
<td><code>borderColor</code></td>
<td><code>GuiFrameSetCtrl</code></td>
</tr>
<tr>
<td><code>borderEnable</code></td>
<td><code>GuiFrameSetCtrl</code></td>
</tr>
<tr>
<td><code>borderMovable</code></td>
<td><code>GuiFrameSetCtrl</code></td>
</tr>
<tr>
<td><code>borderWidth</code></td>
<td><code>GuiFrameSetCtrl</code></td>
</tr>
<tr>
<td><code>bringToFront(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>callOnChildren(string method, string args...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>callOnChildrenNoRecurse(string method, string args...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td>Method/Function</td>
<td>Class/Type</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>columns</td>
<td>GuiFrameSetCtrl</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsIsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>docking</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>frameBorder(int index, string state=&quot;dynamic&quot;)</td>
<td>GuiFrameSetCtrl</td>
</tr>
<tr>
<td>frameMinExtent(int index, int width, int height)</td>
<td>GuiFrameSetCtrl</td>
</tr>
<tr>
<td>frameMovable(int index, string state=&quot;dynamic&quot;)</td>
<td>GuiFrameSetCtrl</td>
</tr>
<tr>
<td>framePadding(int index, RectSpacingI padding)</td>
<td>GuiFrameSetCtrl</td>
</tr>
<tr>
<td>fudgeFactor</td>
<td>GuiFrameSetCtrl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getColumnCount()</td>
<td>GuiFrameSetCtrl</td>
</tr>
<tr>
<td>getColumnOffset(int index)</td>
<td>GuiFrameSetCtrl</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFramePadding(int index)</td>
<td>GuiFrameSetCtrl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRowCount()</td>
<td>GuiFrameSetCtrl</td>
</tr>
<tr>
<td>getRowOffset(int index)</td>
<td>GuiFrameSetCtrl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class/Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>padding</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>removeColumn()</td>
<td>GuiFrameSetCtrl</td>
</tr>
<tr>
<td>removeRow()</td>
<td>GuiFrameSetCtrl</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>rows</td>
<td>GuiFrameSetCtrl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setColumnOffset(int index, int offset)</td>
<td>GuiFrameSetCtrl</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>setFieldType</td>
<td>(string fieldName, string type) SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>(string fieldName, string value, int index=-1) SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>(string fileName) SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden</td>
<td>(bool value=true) SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>(string newInternalName) SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>(bool state=true) SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>(bool state=true) SimObject</td>
</tr>
<tr>
<td>setLocked</td>
<td>(bool value=true) SimObject</td>
</tr>
<tr>
<td>setName</td>
<td>(string newName) SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>(bool value=true) SimObject</td>
</tr>
<tr>
<td>setPosition</td>
<td>(int x, int y) GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal</td>
<td>(int x, int y) GuiControl</td>
</tr>
<tr>
<td>setProfile</td>
<td>(GuiControlProfile profile) GuiControl</td>
</tr>
<tr>
<td>setRowOffset</td>
<td>(int index, int offset) GuiFrameSetCtrl</td>
</tr>
<tr>
<td>setSuperClassNamespace</td>
<td>(string name) SimObject</td>
</tr>
<tr>
<td>setValue</td>
<td>(string value) GuiControl</td>
</tr>
<tr>
<td>setVisible</td>
<td>(bool state=true) GuiControl</td>
</tr>
<tr>
<td>sort</td>
<td>(string callbackFunction) SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>updateSizes</td>
<td>() GuiFrameSetCtrl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
GuiGameListMenuCtrl Member List

This is the complete list of members for GuiGameListMenuCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>activateRow()</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>addRow(string label, string callback, int icon=-1, int yPad=0, bool useHighlightIcon=true, bool enabled=true)</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callbackOnA</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>callbackOnB</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>callbackOnX</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>callbackOnY</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>debugRender</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()透</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRowCount()</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>getRowLabel(int row)</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>getSelectedRow()</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRowEnabled(int row)</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>type</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onChange()</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>remove</code></td>
<td>(SimObject objects...) SimSet</td>
</tr>
<tr>
<td><code>reorderChild</code></td>
<td>(SimObject child1, SimObject child2) SimSet</td>
</tr>
<tr>
<td><code>resize</code></td>
<td>(int x, int y, int width, int height) GuiControl</td>
</tr>
<tr>
<td><code>save</code></td>
<td>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;) SimObject</td>
</tr>
<tr>
<td><code>schedule</code></td>
<td>(float time, string method, string args...) SimObject</td>
</tr>
<tr>
<td><code>setActive</code></td>
<td>(bool state=true) GuiControl</td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td>(bool value=true) SimObject</td>
</tr>
<tr>
<td><code>setCenter</code></td>
<td>(int x, int y) GuiControl</td>
</tr>
<tr>
<td><code>setClassNameSpace</code></td>
<td>(string name) SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td>(bool value=true) SimObject</td>
</tr>
<tr>
<td><code>setExtent</code></td>
<td>(Point2I p) GuiControl</td>
</tr>
<tr>
<td><code>setExtent</code></td>
<td>(S32 width, S32 height) GuiControl</td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td>(string fieldName, string type) SimObject</td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td>(string fieldName, string value, int index=-1) SimObject</td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td>(string fileName) SimObject</td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td>() GuiControl</td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td>(bool value=true) SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td>(string newInternalName) SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td>(bool state=true) SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td>(bool state=true) SimObject</td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td>(bool value=true) SimObject</td>
</tr>
<tr>
<td><code>setName</code></td>
<td>(string newName) SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td>(bool value=true) SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>setPosition(x, y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(x, y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setRowEnabled(row, enabled)</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>setRowLabel(row, label)</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>setSelected(row)</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>setSuperClassNamespace(name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
GuiGameListMenuProfile Member List

This is the complete list of members for **GuiGameListMenuProfile**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>autoSizeHeight</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>autoSizeWidth</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>bevelColorHL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>bevelColorLL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>bitmap</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>border</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>borderColor</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>borderColorHL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>borderColorNA</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>borderThickness</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canKeyFocus</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>category</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>cursorColor</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>fillColor</code></td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td><code>fillColorHL</code></td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td><code>fillColorNA</code></td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td><code>fillColorSEL</code></td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td><code>fontCharset</code></td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td><code>fontColor</code></td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td><code>fontColorHL</code></td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td><code>fontColorLink</code></td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td><code>fontColorLinkHL</code></td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td><code>fontColorNA</code></td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td><code>fontColors</code></td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td><code>fontColorSEL</code></td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td><code>fontSize</code></td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td><code>fontType</code></td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class Name</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td><code>getFieldValue</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>hasBitmapArray</code></td>
<td><code>GuiControlProfile</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>hitAreaLowerRight</code></td>
<td><code>GuiGameListMenuProfile</code></td>
</tr>
<tr>
<td><code>hitAreaUpperLeft</code></td>
<td><code>GuiGameListMenuProfile</code></td>
</tr>
<tr>
<td><code>iconOffset</code></td>
<td><code>GuiGameListMenuProfile</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup</code> (<code>SimGroup</code> <code>group</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField</code> (<code>string</code> <code>fieldName</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code> (<code>string</code> <code>name</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass</code> (<code>string</code> <code>className</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod</code> (<code>string</code> <code>methodName</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>justify</code></td>
<td><code>GuiControlProfile</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>modal</code></td>
<td><code>GuiControlProfile</code></td>
</tr>
<tr>
<td><code>mouseOverSelected</code></td>
<td><code>GuiControlProfile</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>numbersOnly</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>opaque</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>profileForChildren</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>returnTab</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>rowSize</td>
<td>GuiGameListMenuProfile</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method/Property</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>setSuperClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>soundButtonDown</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>soundButtonOver</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tab</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>textOffset</td>
<td>GuiControlProfile</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
**GuiGameListOptionsMenuCtrl Member List**

This is the complete list of members for `GuiGameListOptionsMenuCtrl`, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>accelerator</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>acceptsAsChild(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>activateRow()</code></td>
<td><code>GuiGameListMenuCtrl</code></td>
</tr>
<tr>
<td><code>active</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>add(SimObject objects...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>addGuiControl(GuiControl control)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>addRow(string label, string options, bool wrapOptions, string callback, int icon=-1, int yPad=0, bool enabled=true)</code></td>
<td><code>GuiGameListOptionsMenuCtrl</code></td>
</tr>
<tr>
<td><code>GuiGameListMenuCtrl::addRow(string label, string callback, int icon=-1, int yPad=0, bool useHighlightIcon=true, bool enabled=true)</code></td>
<td><code>GuiGameListMenuCtrl</code></td>
</tr>
<tr>
<td><code>altCommand</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>bringToFront(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>callbackOnA</code></td>
<td><code>GuiGameListMenuCtrl</code></td>
</tr>
<tr>
<td><code>callbackOnB</code></td>
<td><code>GuiGameListMenuCtrl</code></td>
</tr>
<tr>
<td><code>callbackOnX</code></td>
<td><code>GuiGameListMenuCtrl</code></td>
</tr>
<tr>
<td><code>callbackOnY</code></td>
<td><code>GuiGameListMenuCtrl</code></td>
</tr>
<tr>
<td><code>callOnChildren(string method, string args...)</code></td>
<td><code>GuiGameListMenuCtrl</code></td>
</tr>
<tr>
<td><code>callOnChildrenNoRecurse(string</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Method/Function</td>
<td>Class/Type</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder( ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>debugRender</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump( detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getCurrentOption(int row)</code></td>
<td>GuiGameListOptionsCtrl</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getGlobalCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGlobalPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getMinExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getParent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRowCount()</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>getRowLabel(int row)</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>getSelectedRow()</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRowEnabled(int row)</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>makeFirstResponder</td>
<td>boolean isFirst</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onChange()</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method/Function</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>selectOption(int row, string option)</td>
<td>GuiGameListOptionsCtrl</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setOptions(int row, string optionsList)</td>
<td>GuiGameListOptionsCtrl</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setRowEnabled(int row, bool enabled)</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>setRowLabel(int row, string label)</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>setSelected(int row)</td>
<td>GuiGameListMenuCtrl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
GuiGameListOptionsProfile Member List

This is the complete list of members for **GuiGameListOptionsProfile**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>autoSizeHeight</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>autoSizeWidth</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>bevelColorHL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>bevelColorLL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>bitmap</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>border</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>borderColor</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>borderColorHL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>borderColorNA</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>borderThickness</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canKeyFocus</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>category</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>columnSplit</td>
<td>GuiGameListOptionsProfile</td>
</tr>
<tr>
<td>cursorColor</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>fillColor</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fillColorHL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fillColorNA</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fillColorSEL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontCharset</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontColor</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontColorHL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontColorLink</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontColorLinkHL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontColorNAHL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontColors</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontColorSEL</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontSize</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>fontType</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class Name</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getFieldType</code> (string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue</code> (string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getStringWidth</code> (pString)</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hasBitmapArray()</code></td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hitAreaLowerRight</code></td>
<td>GuiGameListMenuProfile</td>
</tr>
<tr>
<td><code>hitAreaUpperLeft</code></td>
<td>GuiGameListMenuProfile</td>
</tr>
<tr>
<td><code>iconOffset</code></td>
<td>GuiGameListMenuProfile</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup</code> (SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField</code> (string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass</code> (string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod</code> (string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>justify</code></td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>mouseOverSelected</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>numbersOnly</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>opaque</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>profileForChildren</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>returnTab</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>rightPad</td>
<td>GuiGameListOptionsProfile</td>
</tr>
<tr>
<td>rowSize</td>
<td>GuiGameListMenuProfile</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>soundButtonDown</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>soundButtonOver</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tab</td>
<td>GuiControlProfile</td>
</tr>
<tr>
<td>textOffset</td>
<td>GuiControlProfile</td>
</tr>
</tbody>
</table>
# GuiGraphCtrl Member List

This is the complete list of members for GuiGraphCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addAutoPlot(int plotId, string variable, int updateFrequency)</td>
<td>GuiGraphCtrl</td>
</tr>
<tr>
<td>addDatum(int plotId, float value)</td>
<td>GuiGraphCtrl</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>centerY</td>
<td>GuiGraphCtrl</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDatum(int plotId, int index)</td>
<td>GuiGraphCtrl</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getGlobalCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGlobalPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getMinExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getParent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getRoot()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getValue()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isActive()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isContainer</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMember(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMouseLocked()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isVisible()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>langTableMod</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>listObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>makeFirstResponder(bool isFirst)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>matchScale(int plotID1, int plotID2,...)</code></td>
<td><code>GuiGraphCtrl</code></td>
</tr>
<tr>
<td><code>minExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>modal</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>onAction()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onActive(bool state)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onAdd()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragEnter(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragExit(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragged(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Function/Method</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>onControlDropped</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onDialogPop</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onDialogPush</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onGainFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onLoseFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onObjectAdded</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onObjectRemoved</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onRemove</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onSleep</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onVisible</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onWake</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>plotColor</code></td>
<td>GuiGraphCtrl</td>
</tr>
<tr>
<td><code>plotInterval</code></td>
<td>GuiGraphCtrl</td>
</tr>
<tr>
<td><code>plotType</code></td>
<td>GuiGraphCtrl</td>
</tr>
<tr>
<td><code>plotVariable</code></td>
<td>GuiGraphCtrl</td>
</tr>
<tr>
<td><code>pointInControl</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>position</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>profile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>pushToBack</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>remove</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>removeAutoPlot</code></td>
<td>GuiGraphCtrl</td>
</tr>
<tr>
<td><code>reorderChild</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>resize</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>save</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setGraphType(int plotId, GuiGraphType graphType)</td>
<td>GuiGraphCtrl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>sort</strong> (string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# GuiHealthBarHud Member List

This is the complete list of members for `GuiHealthBarHud`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td>active</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>altCommand</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td>canSave</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>class</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>className</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>clear()</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>clone()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>command</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>damageFillColor</td>
<td><code>GuiHealthBarHud</code></td>
</tr>
<tr>
<td>deepClone()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>displayEnergy</code></td>
<td>GuiHealthBarHud</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>extent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>fillColor</code></td>
<td>GuiHealthBarHud</td>
</tr>
<tr>
<td><code>findHitControl(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findHitControls(int x, int y, int width, int height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findObjectByInternalName()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>frameColor</code></td>
<td>GuiHealthBarHud</td>
</tr>
<tr>
<td><code>getAspect()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getGlobalCenter()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGlobalPosition()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getMinExtent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getParent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getRoot()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getValue()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isActive()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isContainer</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMember(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMouseLocked()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isVisible()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>langTableMod</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>listObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>makeFirstResponder(bool isFirst)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>minExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>modal</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>onAction()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onActive(bool state)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onAdd()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragEnter(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragExit(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragged(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDropped(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove(SimObject)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pulseRate</td>
<td>GuiHealthBarHud</td>
</tr>
<tr>
<td>pulseThreshold</td>
<td>GuiHealthBarHud</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>(string name)</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setExtent</td>
<td>(Point2I p)</td>
</tr>
<tr>
<td>setExtent</td>
<td>(S32 width, S32 height)</td>
</tr>
<tr>
<td>setFieldType</td>
<td>(string fieldName, string type)</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>(string fieldName, string value, int index=-1)</td>
</tr>
<tr>
<td>setFilename</td>
<td>(string fileName)</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td></td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>()</td>
</tr>
<tr>
<td>setHidden</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setInternalName</td>
<td>(string newInternalName)</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>(bool state=true)</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>(bool state=true)</td>
</tr>
<tr>
<td>setLocked</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setName</td>
<td>(string newName)</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setPosition</td>
<td>(int x, int y)</td>
</tr>
<tr>
<td>setPositionGlobal</td>
<td>(int x, int y)</td>
</tr>
<tr>
<td>setProfile</td>
<td>(GuiControlProfile profile)</td>
</tr>
<tr>
<td>setSuperClassNamespace</td>
<td>(string name)</td>
</tr>
<tr>
<td>setValue</td>
<td>(string value)</td>
</tr>
<tr>
<td>setVisible</td>
<td>(bool state=true)</td>
</tr>
<tr>
<td>showFill</td>
<td></td>
</tr>
<tr>
<td>showFrame</td>
<td></td>
</tr>
<tr>
<td>sort</td>
<td>(string callbackFunction)</td>
</tr>
<tr>
<td>superClass</td>
<td></td>
</tr>
<tr>
<td>tooltip</td>
<td></td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>---------------</td>
<td>------------</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
GuiHealthTextHud Member List

This is the complete list of members for GuiHealthTextHud, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>SimObject</td>
</tr>
<tr>
<td>fillColor</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>frameColor</td>
<td>GuiHealthTextHud</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method/Property</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------</td>
<td>------</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pulseRate</td>
<td>GuiHealthTextHud</td>
</tr>
<tr>
<td>pulseThreshold</td>
<td>GuiHealthTextHud</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setValue(string value)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setVisible(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>showEnergy</code></td>
<td><code>GuiHealthTextHud</code></td>
</tr>
<tr>
<td><code>showFill</code></td>
<td><code>GuiHealthTextHud</code></td>
</tr>
<tr>
<td><code>showFrame</code></td>
<td><code>GuiHealthTextHud</code></td>
</tr>
<tr>
<td><code>showTrueValue</code></td>
<td><code>GuiHealthTextHud</code></td>
</tr>
<tr>
<td><code>sort(string callbackFunction)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>textColor</td>
<td>GuiHealthTextHud</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
<tr>
<td>warningColor</td>
<td>GuiHealthTextHud</td>
</tr>
<tr>
<td>warnThreshold</td>
<td>GuiHealthTextHud</td>
</tr>
</tbody>
</table>
**GuiIconButtonCtrl Member List**

This is the complete list of members for `GuiIconButtonCtrl`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Class Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>autoSize</td>
<td>GuiIconButtonCtrl</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>buttonMargin</td>
<td>GuiIconButtonCtrl</td>
</tr>
<tr>
<td>buttonType</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getld()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIdx(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getText()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>groupNum</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>iconBitmap</td>
<td>GuiIconButtonCtrl</td>
</tr>
<tr>
<td>iconLocation</td>
<td>GuiIconButtonCtrl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class Name</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>makeIconSquare</td>
<td>GuIconButtonCtrl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control,</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Point2I dropPoint)</td>
<td></td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onControlDragExit</td>
<td>GuiControl control, Point2I dropPoint</td>
</tr>
<tr>
<td>onControlDragged</td>
<td>GuiControl control, Point2I dropPoint</td>
</tr>
<tr>
<td>onControlDropped</td>
<td>GuiControl control, Point2I dropPoint</td>
</tr>
<tr>
<td>onDialogPop</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDoubleClick</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onGainFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onMouseDown</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseDragged</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseEnter</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseLeave</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseUp</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onObjectAdded</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onRightClick</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onSleep</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>performClick</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td><code>profile</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>pushToBack(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>remove(SimObject objects...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>reorderChild(SimObject child1, SimObject child2)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>resetState()</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setBitmap(string buttonFilename)</code></td>
<td><code>GuiIconButtonCtrl</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setStateOn(bool isOn=true)</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setText(string text)</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>setTextID(string id)</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sizeIconToButton</td>
<td>GuiIconButtonCtrl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>text</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>textID</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>textLocation</td>
<td>GuiIconButtonCtrl</td>
</tr>
<tr>
<td>textMargin</td>
<td>GuiIconButtonCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>useMouseEvents</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
# GuiInputCtrl Member List

This is the complete list of members for GuiInputCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>SimObject</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullName()</td>
<td>SimSet</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>lockMouse</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------------------------------------</td>
</tr>
<tr>
<td>onInputEvent(String device, String action, bool state)</td>
<td>GuiInputCtrl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onMouseDown(U8 modifier, Point2I mousePoint, U8 mouseClickCount)</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>onMouseDragged(U8 modifier, Point2I mousePoint, U8 mouseClickCount)</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>onMouseEnter(U8 modifier, Point2I mousePoint, U8 mouseClickCount)</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>onMouseLeave(U8 modifier, Point2I mousePoint, U8 mouseClickCount)</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>onMouseMove(U8 modifier, Point2I mousePoint, U8 mouseClickCount)</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>onMouseUp(U8 modifier, Point2I mousePoint, U8 mouseClickCount)</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onRightMouseDown(U8 modifier, Point2I mousePoint, U8 mouseClickCount)</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>onRightMouseDragged(U8 modifier, Point2I mousePoint, U8 mouseClickCount)</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>onRightMouseUp(U8 modifier, Point2I mousePoint, U8 mouseClickCount)</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
profile

**pushToBack**(SimObject obj) | GuiControl
---|---
**remove**(SimObject objects...) | SimSet
**reorderChild**(SimObject child1, SimObject child2) | SimSet
**resize**(int x, int y, int width, int height) | GuiControl
**save**(string fileName, bool selectedOnly=false, string preAppendString="") | SimObject
**schedule**(float time, string method, string args...) | SimObject
**setActive**(bool state=true) | GuiControl
**setCanSave**(bool value=true) | SimObject
**setCenter**(int x, int y) | GuiControl
**setClassNamespace**(string name) | SimObject
**setEditorOnly**(bool value=true) | SimObject
**setExtent**(Point2I p) | GuiControl
**setExtent**(S32 width, S32 height) | GuiControl
**setFieldType**(string fieldName, string type) | SimObject
**setFieldValue**(string fieldName, string value, int index=-1) | SimObject
**setFilename**(string fileName) | SimObject
**setFirstResponder** | GuiControl
**setFirstResponder**() | GuiControl
**setHidden**(bool value=true) | SimObject
**setInternalName**(string newInternalName) | SimObject
**setIsExpanded**(bool state=true) | SimObject
**setIsSelected**(bool state=true) | SimObject
<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
GuiListBoxCtrl Member List

This is the complete list of members for GuiListBoxCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addFilteredItem(string newItem)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>allowMultipleSelections</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clearColor(int index)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>clearItems()</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>clearSelection()</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>colorBullet</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controllersChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>deleteItem(int itemIndex)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>doMirror()</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findItemText(string findText, bool bCaseSensitive=false)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>fitParentWidth</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getItemCount()</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>getItemObject(int index)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>getItemText(int index)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>getLastClickItem()</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td><code>getSelCount()</code></td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td><code>getSelectedItem()</code></td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td><code>getSelectedItems()</code></td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getValue()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>insertItem(string text, int index)</code></td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isActive</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isContainer</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMember(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMouseLocked()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isObjectMirrored(string indexIdString)</code></td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isVisible()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>langTableMod</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>listObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>makeFirstResponder(bool isFirst)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>makeNameCallback</code></td>
<td><code>GuiListBoxCtrl</code></td>
</tr>
<tr>
<td><code>minExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>mirrorSet</code></td>
<td><code>GuiListBoxCtrl</code></td>
</tr>
<tr>
<td><code>modal</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>onAction()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onActive(bool state)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onAdd()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onClearSelection()</code></td>
<td><code>GuiListBoxCtrl</code></td>
</tr>
<tr>
<td><code>onControlDragEnter(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragExit(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragged(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDropped(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onDeleteKey()</code></td>
<td><code>GuiListBoxCtrl</code></td>
</tr>
<tr>
<td><code>onDialogPop()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onDialogPush()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onDoubleClick()</code></td>
<td><code>GuiListBoxCtrl</code></td>
</tr>
<tr>
<td><code>onGainFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onLoseFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onMouseDragged()</code></td>
<td><code>GuiListBoxCtrl</code></td>
</tr>
<tr>
<td><code>onMouseUp(string itemHit, string mouseClickCount)</code></td>
<td><code>GuiListBoxCtrl</code></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>onObjectAdded</td>
<td>(SimObject object)</td>
</tr>
<tr>
<td>onObjectRemoved</td>
<td>(SimObject object)</td>
</tr>
<tr>
<td>onRemove</td>
<td></td>
</tr>
<tr>
<td>onSelect</td>
<td>(string index, string itemText)</td>
</tr>
<tr>
<td>onSleep</td>
<td></td>
</tr>
<tr>
<td>onUnselect</td>
<td>(string index, string itemText)</td>
</tr>
<tr>
<td>onVisible</td>
<td>(bool state)</td>
</tr>
<tr>
<td>onWake</td>
<td></td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl</td>
<td>(int x, int y)</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack</td>
<td>(SimObject obj)</td>
</tr>
<tr>
<td>remove</td>
<td>(SimObject objects...)</td>
</tr>
<tr>
<td>removeFilteredItem</td>
<td>(string itemName)</td>
</tr>
<tr>
<td>reorderChild</td>
<td>(SimObject child1, SimObject child2)</td>
</tr>
<tr>
<td>resize</td>
<td>(int x, int y, int width, int height)</td>
</tr>
<tr>
<td>save</td>
<td>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
</tr>
<tr>
<td>schedule</td>
<td>(float time, string method, string args...)</td>
</tr>
<tr>
<td>setActive</td>
<td>(bool state=true)</td>
</tr>
<tr>
<td>setCanSave</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setCenter</td>
<td>(int x, int y)</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>(string name)</td>
</tr>
<tr>
<td>setCurSel</td>
<td>(int indexId)</td>
</tr>
<tr>
<td>setCurSelRange</td>
<td>(int indexStart, int indexStop=999999)</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>setEditorOnly (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setItemColor(int index, ColorF color)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>setItemText(int index, string newtext)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>setItemTooltip(int index, string text)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setMultipleSelection(bool allowMultSelections)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSelected(int index, bool setSelected=true)</td>
<td>GuiListBoxCtrl</td>
</tr>
<tr>
<td>setSuperClassNameNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>--------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## GuiMenuBar Member List

This is the complete list of members for GuiMenuBar, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>addMenu(string menuText, int menuId)</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td>addMenuItem(string targetMenu=NULL, string menuItemText='', int menuItemId=0, string accelerator=NULL, int checkGroup=-1)</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td>addSubmenuItem(string menuTarget, string menuItem, string submenuItemText, int submenuItemId, string accelerator, int checkGroup)</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>clearFirstResponder(boold_ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clearMenuItems(string menuTarget)</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td>clearMenus(int param1, int param2)</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td>clearSubmenuItems(string menuTarget, string menuitem)</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getGlobalCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGlobalPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getMinExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getParent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getRoot()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getValue</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>is ACTIVE</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isChildofGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isContainer</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMember(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMouseLocked()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isVisible()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>langTableMod</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>listObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>makeFirstResponder(bool isFirst)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>minExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>modal</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>onAction()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onActive(bool state)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onAdd()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>onControlDragEnter</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onMenuItemSelect</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td>onMenuSelect</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td>onMouseInMenu</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td>onObjectAdded</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSubmenuSelect</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td>onVisible</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>padding</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(x, y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>pushToBack(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>remove(SimObject objects...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>removeMenu(string menuTarget)</code></td>
<td><code>GuiMenuBar</code></td>
</tr>
<tr>
<td><code>removeMenuItems(string menuTarget, string menuItemTarget)</code></td>
<td><code>GuiMenuBar</code></td>
</tr>
<tr>
<td><code>reorderChild(SimObject child1, SimObject child2)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setCheckmarkBitmapIndex(int bitmapIndex)</code></td>
<td><code>GuiMenuBar</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code> setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
</tbody>
</table>

Note: The table lists various functions and their corresponding classes in a structured format.
<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>setLocked</code></td>
<td>Set whether the object is locked</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setMenuBitmapIndex</code></td>
<td>Set the menu bitmap index for the target menu</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td><code>setMenuItemBitmap</code></td>
<td>Set the menu item bitmap for the target menu and item</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td><code>setMenuItemChecked</code></td>
<td>Set the menu item as checked</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td><code>setMenuItemEnable</code></td>
<td>Set the menu item as enabled</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td><code>setMenuItemSubmenuState</code></td>
<td>Set the menu item as sub menu state</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td><code>setMenuItemText</code></td>
<td>Set the text of the menu item</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td><code>setMenuItemVisible</code></td>
<td>Set the visibility of the menu item</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td><code>setMenuMargins</code></td>
<td>Set the margins of the menu</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td><code>setMenuText</code></td>
<td>Set the text of the menu</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td><code>setMenuVisible</code></td>
<td>Set the visibility of the menu</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td><code>setName</code></td>
<td>Set the name of the object</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td>Set whether name change is allowed</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setPosition</code></td>
<td>Set the position of the object</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setPositionGlobal</code></td>
<td>Set the global position of the object</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setProcessTicks</code></td>
<td>Enable or disable process ticks</td>
<td>GuiTickCtrl</td>
</tr>
<tr>
<td><code>setProfile</code></td>
<td>Set the profile of the control</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setSubmenuItemChecked</code></td>
<td>Set the sub menu item as checked</td>
<td>GuiMenuBar</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td>Set the superclass namespace</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setValue</code></td>
<td>Set the value of the object</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
<td></td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
<td></td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
<td></td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
<td></td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
<td></td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
<td></td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
<td></td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
<td></td>
</tr>
</tbody>
</table>
# GuiMessageVectorCtrl Member List

This is the complete list of members for **GuiMessageVectorCtrl**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>allowedMatches</td>
<td>GuiMessageVectorCtrl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>attach(MessageVector item)</td>
<td>GuiMessageVectorCtrl</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>controlsChild(GuiControl control)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>detach()</code></td>
<td>GuiMessageVectorCtrl</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>extent</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findHitControl(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findHitControls(int x, int y, int width, int height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findObjectByInternalName(string internalName, bool searchChildren=false)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getAspects()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>getFieldType</code> (string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue</code> (string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getGlobalCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGlobalPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getMinExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getObjectIndex</code> (SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getParent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getRoot()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getValue</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isActive</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isChildOfGroup</code> (SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Returns</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>lineContinuedIndex</td>
<td>GuiMessageVectorCtrl</td>
</tr>
<tr>
<td>lineSpacing</td>
<td>GuiMessageVectorCtrl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>matchColor</td>
<td>GuiMessageVectorCtrl</td>
</tr>
<tr>
<td>maxColorIndex</td>
<td>GuiMessageVectorCtrl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control,)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>schedule</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setActive</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code> setHidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code> setIsExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setPosition</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setPositionGlobal</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setProfile</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setValue</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setVisible</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------</td>
</tr>
<tr>
<td><code>sort</code> (string <code>callbackFunction</code>)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# GuiMLTextCtrl Member List

This is the complete list of members for GuiMLTextCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Member Function</th>
<th>Class Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl</td>
<td>GuiControl</td>
</tr>
<tr>
<td>addText</td>
<td>GuiMLTextCtrl</td>
</tr>
<tr>
<td>allowColorChars</td>
<td>GuiMLTextCtrl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront</td>
<td>SimSet</td>
</tr>
<tr>
<td>call</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>deniedSound</td>
<td>GuiMLTextCtrl</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>forceReflow()</td>
<td>GuiMLTextCtrl</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getText()</td>
<td>GuiMLTextCtrl</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>lineSpacing</td>
<td>GuiMLTextCtrl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>maxChars</td>
<td>GuiMLTextCtrl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Docstring</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td><code>onControlDragged</code></td>
<td><code>GuiControl control, Point2I dropPoint</code></td>
</tr>
<tr>
<td><code>onControlDropped</code></td>
<td><code>GuiControl control, Point2I dropPoint</code></td>
</tr>
<tr>
<td><code>onDialogPop()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onDialogPush()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onGainFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onLoseFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onObjectAdded(object)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>onObjectRemoved(object)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>onRemove()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onResize(width, maxY)</code></td>
<td><code>GuiMLTextCtrl</code></td>
</tr>
<tr>
<td><code>onSleep()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onURL(url)</code></td>
<td><code>GuiMLTextCtrl</code></td>
</tr>
<tr>
<td><code>onVisible(state)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onWake()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>pointInControl(x, y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>position</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>profile</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>pushToBack(obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>remove(objects)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>reorderChild(child1, child2)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>resize(x, y, width, height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>save(fileName, selectedOnly=false, preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(time, method, args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>scrollToBottom()</code></td>
<td><code>GuiMLTextCtrl</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>scrollToTag(int tagID)</td>
<td>GuiMLTextCtrl</td>
</tr>
<tr>
<td>scrollToTop(int param1, int param2)</td>
<td>GuiMLTextCtrl</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setAlpha(float alphaVal)</td>
<td>GuiMLTextCtrl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCursorPosition(int newPos)</td>
<td>GuiMLTextCtrl</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>setText</code></td>
<td><code>GuiMLTextCtrl</code></td>
</tr>
<tr>
<td><code>setValue</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setVisible</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>sort</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>text</code></td>
<td><code>GuiMLTextCtrl</code></td>
</tr>
<tr>
<td><code>tooltip</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>tooltipProfile</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>useURLMouseCursor</code></td>
<td><code>GuiMLTextCtrl</code></td>
</tr>
<tr>
<td><code>variable</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>vertSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>visible</code></td>
<td><code>GuiControl</code></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
This is the complete list of members for **GuiMLTextEditCtrl**, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>addText(string text, bool reformat)</td>
<td>GuiMLTextCtrl</td>
</tr>
<tr>
<td>allowColorChars</td>
<td>GuiMLTextCtrl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecuse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>deniedSound</td>
<td>GuiMLTextCtrl</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>escapeCommand</td>
<td>GuiMLTextEditCtrl</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>forceReflow()</td>
<td>GuiMLTextCtrl</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td><code>getFieldType</code> (string fieldName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue</code> (string fieldName, int index=-1)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getFullCount</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getGlobalCenter</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGlobalPosition</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getMinExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getParent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getPosition</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getRandom</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getRoot</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getText()</code></td>
<td><code>GuiMLTextCtrl</code></td>
</tr>
<tr>
<td><code>getValue</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isActive</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>lineSpacing</td>
<td>GuiMLTextCtrl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>maxChars</td>
<td>GuiMLTextCtrl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>onControlDragged</td>
<td>GuiControl dropPoint</td>
</tr>
<tr>
<td>onControlDropped</td>
<td>GuiControl dropPoint</td>
</tr>
<tr>
<td>onDialogPop</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded</td>
<td>SimSet object</td>
</tr>
<tr>
<td>onObjectRemoved</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onResize</td>
<td>GuiMLTextCtrl width, maxY</td>
</tr>
<tr>
<td>onSleep</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onURL</td>
<td>GuiMLTextCtrl url</td>
</tr>
<tr>
<td>onVisible</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td></td>
<td>fileName, bool selectedOnly=false,</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>schedule</code></td>
<td>(float time, string method, string args...)</td>
</tr>
<tr>
<td><code>scrollToBottom</code></td>
<td>GuiMLTextCtrl</td>
</tr>
<tr>
<td><code>scrollToTag</code></td>
<td>int tagID</td>
</tr>
<tr>
<td><code>scrollToTop</code></td>
<td>int param1, int param2</td>
</tr>
<tr>
<td><code>setActive</code></td>
<td>bool state=true</td>
</tr>
<tr>
<td><code>setAlpha</code></td>
<td>float alphaVal</td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td>bool value=true</td>
</tr>
<tr>
<td><code>setCenter</code></td>
<td>int x, int y</td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td>string name</td>
</tr>
<tr>
<td><code>setCursorPosition</code></td>
<td>int newPos</td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td>bool value=true</td>
</tr>
<tr>
<td><code>setExtent</code></td>
<td>Point2I p</td>
</tr>
<tr>
<td><code>setExtent</code></td>
<td>S32 width, S32 height</td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td>string fieldName, string type</td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td>string fieldName, string value, int index=-1</td>
</tr>
<tr>
<td><code>setName</code></td>
<td>string fileName</td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td>bool value=true</td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td>()</td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td>bool value=true</td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td>string newInternalName</td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td>bool state=true</td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td>bool state=true</td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td>bool value=true</td>
</tr>
<tr>
<td><code>setName</code></td>
<td>string newGroupName</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td>bool value=true</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setText(string text)</code></td>
<td><code>GuiMLTextCtrl</code></td>
</tr>
<tr>
<td><code>setValue(string value)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setVisible(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>sort(string callbackFunction)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>text</code></td>
<td><code>GuiMLTextCtrl</code></td>
</tr>
<tr>
<td><code>tooltip</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>tooltipProfile</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>useURLMouseCursor</code></td>
<td><code>GuiMLTextCtrl</code></td>
</tr>
<tr>
<td><code>variable</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>vertSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>visible</code></td>
<td><code>GuiControl</code></td>
</tr>
</tbody>
</table>
## GuiMouseEventCtrl Member List

This is the complete list of members for `GuiMouseEventCtrl`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>--------------------------------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>extent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControl(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControls(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findObjectByInternalName(string internalName, bool searchChildren=false)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getAspect()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td><code>isMember(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMouseLocked()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isVisible()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>langTableMod</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>listObjects()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>lockMouse</code></td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td><code>makeFirstResponder(bool isFirst)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>minExtent</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>modal</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>onAction()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onActive(bool state)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onAdd()</code></td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
| `onControlDragEnter(GuiControl control,
Point2I dropPoint)`                           | GuiControl                                       |
| `onControlDragExit(GuiControl control,
Point2I dropPoint)`                           | GuiControl                                       |
| `onControlDragged(GuiControl control,
Point2I dropPoint)`                           | GuiControl                                       |
| `onControlDropped(GuiControl control,
Point2I dropPoint)`                           | GuiControl                                       |
<p>| <code>onDialogPop()</code>                               | GuiControl                                       |
| <code>onDialogPush()</code>                              | GuiControl                                       |
| <code>onGainFirstResponder()</code>                      | GuiControl                                       |</p>
<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onMouseDown(U8 modifier, Point2I mousePoint, U8 mouseClickCount)</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>onMouseDragged(U8 modifier, Point2I mousePoint, U8 mouseClickCount)</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>onMouseEnter(U8 modifier, Point2I mousePoint, U8 mouseClickCount)</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>onMouseLeave(U8 modifier, Point2I mousePoint, U8 mouseClickCount)</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>onMouseMove(U8 modifier, Point2I mousePoint, U8 mouseClickCount)</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>onMouseUp(U8 modifier, Point2I mousePoint, U8 mouseClickCount)</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onRightMouseDown(U8 modifier, Point2I mousePoint, U8 mouseClickCount)</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>onRightMouseDragged(U8 modifier, Point2I mousePoint, U8 mouseClickCount)</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>onRightMouseUp(U8 modifier, Point2I mousePoint, U8 mouseClickCount)</td>
<td>GuiMouseEventCtrl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>pushToBack(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>remove(SimObject objects...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>reorderChild(SimObject child1, SimObject child2)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code> setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
This is the complete list of members for `GuiObjectView`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>accelerator</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>acceptsAsChild(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>active</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>add(SimObject objects...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>addGuiControl(GuiControl control)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>altCommand</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>anchorBottom</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>anchorLeft</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>anchorRight</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>anchorTop</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>animSequence</code></td>
<td><code>GuiObjectView</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>bringToFront(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>calculateViewDistance(float radius)</code></td>
<td><code>GuiTSCtrl</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>callOnChildren(string method, string args...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>callOnChildrenNoRecurse(string method, string args...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>cameraSpeed</code></td>
<td><code>GuiObjectView</code></td>
</tr>
<tr>
<td><code>cameraZRot</code></td>
<td><code>GuiTSCtrl</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>clear()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>clearFirstResponder(bool ignored=false)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>command</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>controlsChild(GuiControl control)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>docking</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>extent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControl(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControls(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findObjectByInternalName(string internalName, bool searchChildren=false)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>forceFOV</code></td>
<td><code>GuiTSCtrl</code></td>
</tr>
<tr>
<td><code>getAspect()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getCameraSpeed()</code></td>
<td><code>GuiObjectView</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getModel()</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>getMountedModel()</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>getMountSkin(int param1, int param2)</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getOrbitDistance()</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>getSkin()</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getWorldToScreenScale()</td>
<td>GuiTSCtrl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>lightAmbient</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>lightColor</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>lightDirection</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(boo/<em>isFirst</em>/)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>maxOrbitDiststance</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minOrbitDiststance</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>mountedNode</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>mountedShapeFile</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>mountedSkin</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onMouseEnter()</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>onMouseLeave()</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class/Module</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>onObjectAdded</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>orbitDistance</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>padding</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>project(Point3F worldPosition)</td>
<td>GuiTSCtrl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reflectPriority</td>
<td>GuiTSCtrl</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCameraSpeed(float factor)</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLightAmbient(ColorF color)</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>setLightColor(ColorF color)</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>setLightDirection(Point3F direction)</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setModel(string shapeName)</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>setMount(string shapeName, string mountNodeIndexOrName)</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>setMountedModel(string shapeName)</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>setMountSkin(string skinName)</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setOrbitDistance(float distance)</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSeq(string indexOrName)</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>setSkin</code> (string skinName)</td>
<td>GuiObjectView</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)`</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setValue</code> (string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setVisible</code> (bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>shapeFile</code></td>
<td>GuiObjectView</td>
</tr>
<tr>
<td><code>skin</code></td>
<td>GuiObjectView</td>
</tr>
<tr>
<td><code>sort</code> (string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>tooltip</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>tooltipProfile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>unproject</code> (Point3F screenPosition)</td>
<td>GuiTSCtrl</td>
</tr>
<tr>
<td><code>variable</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>vertSizing</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>visible</code></td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
GuiPaneControl Member List

This is the complete list of members for GuiPaneControl, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>barBehindText</td>
<td>GuiPaneControl</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecursive(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>caption</td>
<td>GuiPaneControl</td>
</tr>
<tr>
<td>captionID</td>
<td>GuiPaneControl</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>collapsable</td>
<td>GuiPaneControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getGlobalCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGlobalPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getMinExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getParent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getRoot()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getValue()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isActive()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isContainer</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>onDialogPop()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onDialogPush()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onGainFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onLoseFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onObjectAdded(SimObject object)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onObjectRemoved(SimObject object)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onRemove()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onSleep()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onVisible(bool state)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onWake()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>pointInControl(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>position</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>profile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>pushToBack(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>remove(SimObject objects...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>reorderChild(SimObject child1, SimObject child2)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCollapsed(bool collapse)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
**GuiPanel Member List**

This is the complete list of members for **GuiPanel**, including all inherited members.

<table>
<thead>
<tr>
<th>Member Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>anchorBottom</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorLeft</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorRight</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorTop</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>docking</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>get Aspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>get CanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>get Center()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>get Class Name()</td>
<td>SimObject</td>
</tr>
<tr>
<td>get Class Namespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>get Debug Info()</td>
<td>SimObject</td>
</tr>
<tr>
<td>get Declaration Line()</td>
<td>SimObject</td>
</tr>
<tr>
<td>get Dynamic Field(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>get Dynamic Field Count()</td>
<td>SimObject</td>
</tr>
<tr>
<td>get Extent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>get Field(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>getFieldType</code></td>
<td>(string fieldName)</td>
</tr>
<tr>
<td><code>getFieldValue</code></td>
<td>(string fieldName, int index=-1)</td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td></td>
</tr>
<tr>
<td><code>getFirstResponder</code></td>
<td></td>
</tr>
<tr>
<td><code>getFullCount</code></td>
<td></td>
</tr>
<tr>
<td><code>getGlobalCenter</code></td>
<td></td>
</tr>
<tr>
<td><code>getGlobalPosition</code></td>
<td></td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td></td>
</tr>
<tr>
<td><code>getId</code></td>
<td></td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td></td>
</tr>
<tr>
<td><code>getMinExtent</code></td>
<td></td>
</tr>
<tr>
<td><code>getName</code></td>
<td></td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td></td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td></td>
</tr>
<tr>
<td><code>getParent</code></td>
<td></td>
</tr>
<tr>
<td><code>getPosition</code></td>
<td></td>
</tr>
<tr>
<td><code>getRandom</code></td>
<td></td>
</tr>
<tr>
<td><code>getRoot</code></td>
<td></td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td></td>
</tr>
<tr>
<td><code>getValue</code></td>
<td></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td></td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td></td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td></td>
</tr>
<tr>
<td><code>isActive</code></td>
<td></td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td></td>
</tr>
<tr>
<td><code>isContainer</code></td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>onControlDropped</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onDialogPop()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onDialogPush()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onGainFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onLoseFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onObjectAdded</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onObjectRemoved</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onRemove()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onSleep()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onVisible(bool state)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onWake()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>padding</code></td>
<td>GuiContainer</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>pointInControl(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>position</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>profile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>pushToBack(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>remove(SimObject objects...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>reorderChild(SimObject child1, SimObject child2)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setValue(string value)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setVisible(bool state=true)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>sort(string callbackFunction)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>tooltip</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>tooltipProfile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>variable</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>-------------</td>
<td>------------</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
**GuiPopUpMenuCtrl Member List**

This is the complete list of members for `GuiPopUpMenuCtrl`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(string name, int idNum, int scheme=0)</td>
<td>GuiPopUpMenuCtrl</td>
</tr>
<tr>
<td>GuiTextCtrl::add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>addScheme(int id, ColorI fontColor, ColorI fontColorHL, ColorI fontColorSEL)</td>
<td>GuiPopUpMenuCtrl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>anchorBottom</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorLeft</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorRight</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorTop</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bitmap</td>
<td>GuiPopUpMenuCtrl</td>
</tr>
<tr>
<td>bitmapBounds</td>
<td>GuiPopUpMenuCtrl</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td><code>changeTextById(id, string text)</code></td>
<td><code>GuiPopUpMenuCtrl</code></td>
</tr>
<tr>
<td><code>clear()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clear()</code></td>
<td><code>GuiPopUpMenuCtrl</code></td>
</tr>
<tr>
<td><code>clearEntry(entry)</code></td>
<td><code>GuiPopUpMenuCtrl</code></td>
</tr>
<tr>
<td><code>clearFirstResponder(ignored=false)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>command()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>containsChild(control)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>docking()</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>dump(detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>extent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControl(x, y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControls(x, y, width, height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findObjectByInternalName(internalName, searchChildren=false)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>findText(string text)</code></td>
<td><code>GuiPopUpMenuCtrl</code></td>
</tr>
<tr>
<td><code>forceClose()</code></td>
<td><code>GuiPopUpMenuCtrl</code></td>
</tr>
<tr>
<td><code>forceOnAction()</code></td>
<td><code>GuiPopUpMenuCtrl</code></td>
</tr>
<tr>
<td><code>getAspect()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>Function Name</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>getRoot()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getSelected</code></td>
<td><code>GuiPopUpMenuCtrl</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getText</code></td>
<td><code>GuiPopUpMenuCtrl</code></td>
</tr>
<tr>
<td><code>getTextById(int id)</code></td>
<td><code>GuiPopUpMenuCtrl</code></td>
</tr>
<tr>
<td><code>getValue</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isActive</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isContainer</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMember(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMouseLocked()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isVisible()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>langTableMod</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>listObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>maxLength</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>maxPopupHeight</td>
<td>GuiPopUpMenuCtrl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove(SimObject object)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td><code>padding</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>pointInControl(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>position</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>profile</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>pushToBack(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>remove(SimObject objects...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>reorderChild(SimObject child1, SimObject child2)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>replaceText(bool doReplaceText)</code></td>
<td><code>GuiPopUpMenuCtrl</code></td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>reverseTextList</code></td>
<td><code>GuiPopUpMenuCtrl</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>sbUsesNAColor</code></td>
<td><code>GuiPopUpMenuCtrl</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEnumContent(string class, string enum)</code></td>
<td><code>GuiPopUpMenuCtrl</code></td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value,</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td><code>int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>SimObject::setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>SimObject::setFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>SimObject::setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>SimObject::setFirstResponder([scriptCallback=true])</code></td>
<td><code>GuiPopUpMenuCtrl</code></td>
</tr>
<tr>
<td><code>SimObject::setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>SimObject::setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>SimObject::setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>SimObject::setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>SimObject::setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>SimObject::setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>SimObject::setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>SimObject::setNoneSelected</code></td>
<td><code>GuiPopUpMenuCtrl</code></td>
</tr>
<tr>
<td><code>SimObject::setPosition(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>SimObject::setPositionGlobal(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>SimObject::setProfile(GuiControlProfile profile)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>SimObject::setSelected(int id,[scriptCallback=true])</code></td>
<td><code>GuiPopUpMenuCtrl</code></td>
</tr>
<tr>
<td><code>SimObject::setSuperClassNameSpace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>SimObject::setText(string text)</code></td>
<td><code>GuiTextCtrl</code></td>
</tr>
<tr>
<td><code>SimObject::setTextID(string textID)</code></td>
<td><code>GuiTextCtrl</code></td>
</tr>
<tr>
<td><code>SimObject::setValue(string value)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>SimObject::setVisible(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>GuiPopUpMenuCtrl::size</code></td>
<td><code>GuiPopUpMenuCtrl</code></td>
</tr>
<tr>
<td><code>GuiPopUpMenuCtrl::sort(string callbackFunction)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>GuiPopUpMenuCtrl::sortID</code></td>
<td><code>GuiPopUpMenuCtrl</code></td>
</tr>
<tr>
<td><code>SimObject::superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Column</td>
<td>Value</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td>text</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>textID</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
GuiPopUpMenuCtrlEx Member List

This is the complete list of members for **GuiPopUpMenuCtrlEx**, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(string name, S32 idNum, S32 scheme=0)</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>add(string name, int idNum, int scheme=0)</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>GuiTextCtrl::add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addCategory(string text)</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>addScheme(int id, ColorI fontColor, ColorI fontColorHL, ColorI fontColorSEL)</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>anchorBottom</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorLeft</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorRight</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorTop</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bitmap</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>bitmapBounds</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method,</td>
<td>SimSet</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>string args...</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>clearEntry(S32 entry)</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>docking</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>findText(string text)</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>forceClose()</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>forceOnAction()</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassname()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getColorById()</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getParent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getRoot()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getSelected()</code></td>
<td><code>GuiPopUpMenuCtrlEx</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getText()</code></td>
<td><code>GuiPopUpMenuCtrlEx</code></td>
</tr>
<tr>
<td><code>getTextById(int id)</code></td>
<td><code>GuiPopUpMenuCtrlEx</code></td>
</tr>
<tr>
<td><code>getValue()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hotTrackCallback</code></td>
<td><code>GuiPopUpMenuCtrlEx</code></td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isActive()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isContained</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMember(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMouseLocked()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>maxLength</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>maxPopupHeight</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>padding</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>replaceText</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>reverseTextList</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>sbUsesNAColor</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td><code>setEnumContent</code></td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNoneSelected(int param)</code></td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setSelected(bool scriptCallback=true)</code></td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td><code>setSelected(int id, bool scriptCallback=true)</code></td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setText(string text)</code></td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td><code>setTextID(string textID)</code></td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td><code>setValue(string value)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setVisible(bool state=true)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>size</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>sort()</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>GuiTextCtrl::sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>sortID()</td>
<td>GuiPopUpMenuCtrlEx</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>text</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>textID</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
GuiProgressBitmapCtrl Member List

This is the complete list of members for GuiProgressBitmapCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>anchorBottom</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorLeft</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorRight</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorTop</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bitmap</td>
<td>GuiProgressBitmapCtrl</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>docking</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>getExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>maxLength</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>onControlDragEnter</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake</td>
<td>GuiControl</td>
</tr>
<tr>
<td>padding</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
save(string fileName, bool selectedOnly=false, string preAppendString="")

schedule(float time, string method, string args...)

setActive(bool state=true)

setBitmap(string filename)

setCanSave(bool value=true)

setCenter(int x, int y)

setClassNamespace(string name)

setEditorOnly(bool value=true)

setExtent(Point2I p)

setExtent(S32 width, S32 height)

setFieldType(string fieldName, string type)

setFieldValue(string fieldName, string value, int index=-1)

setFilename(string fileName)

setFirstResponder()

setHidden(bool value=true)

setInternalName(string newName)

setNameChangeAllowed(bool value=true)

setPosition(int x, int y)

setPositionGlobal(int x, int y)

setProfile(GuiControlProfile profile)
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td>Sets the superclass namespace of an object.</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setText</code></td>
<td>Sets the text of a text control.</td>
<td><code>GuiTextCtrl</code></td>
</tr>
<tr>
<td><code>setTextID</code></td>
<td>Sets the text ID of a text control.</td>
<td><code>GuiTextCtrl</code></td>
</tr>
<tr>
<td><code>setValue</code></td>
<td>Sets the value of a control.</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setVisible</code></td>
<td>Sets the visibility of a control.</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>sort</code></td>
<td>Sorts objects based on a callback function.</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>Returns the superclass of an object.</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>text</code></td>
<td>Returns the text of a text control.</td>
<td><code>GuiTextCtrl</code></td>
</tr>
<tr>
<td><code>textID</code></td>
<td>Returns the text ID of a text control.</td>
<td><code>GuiTextCtrl</code></td>
</tr>
<tr>
<td><code>tooltip</code></td>
<td>Returns the tooltip of a control.</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>tooltipProfile</code></td>
<td>Returns the tooltip profile of a control.</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>variable</code></td>
<td>Returns a variable control.</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>vertSizing</code></td>
<td>Returns the vertical sizing of a control.</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>visible</code></td>
<td>Returns the visibility of a control.</td>
<td><code>GuiControl</code></td>
</tr>
</tbody>
</table>
GuiProgressCtrl Member List

This is the complete list of members for GuiProgressCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>anchorBottom</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorLeft</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorRight</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorTop</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td><code>controlsChild(GuiControl control)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>docking</code></td>
<td>GuiContainer</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>extent</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findHitControl(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findHitControls(int x, int y, int width, int height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findObjectByInternalName(string internalName, bool searchChildren=false)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getAspect()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getGlobalCenter()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGlobalPosition()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getMinExtent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getParent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getRoot()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getValue()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isActive()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isContainer</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMember(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMouseLocked()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isVisible()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>langTableMod</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>listObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>makeFirstResponder(bool isFirst)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>margin</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>maxLength</code></td>
<td><code>GuiTextCtrl</code></td>
</tr>
<tr>
<td><code>minExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>modal</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>onAction()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onActive(bool state)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onAdd()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragEnter(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragExit(GuiControl control, Point2I dropPoint)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>onControlDragged</strong> (GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>onControlDropped</strong> (GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>onDialogPop</strong> ()</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>onDialogPush</strong> ()</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>onGainFirstResponder</strong> ()</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>onLoseFirstResponder</strong> ()</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>onObjectAdded</strong> (SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td><strong>onObjectRemoved</strong> (SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td><strong>onRemove</strong> ()</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>onSleep</strong> ()</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>onVisible</strong> (bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>onWake</strong> ()</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>padding</strong></td>
<td>GuiContainer</td>
</tr>
<tr>
<td><strong>parentGroup</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>persistentId</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>pointInControl</strong> (int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>position</strong></td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>profile</strong></td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>pushToBack</strong> (SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td><strong>remove</strong> (SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td><strong>reorderChild</strong> (SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td><strong>resize</strong> (int x, int y, int width, int height)</td>
<td>SimSet</td>
</tr>
<tr>
<td><strong>save</strong> (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>schedule</strong> (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setActive</strong> (bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>setCanSave</strong> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setText(string text)</code></td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td><code>setTextID(string textID)</code></td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td><code>setValue(string value)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setVisible(bool state=true)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>sort(string callbackFunction)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>text</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>textID</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
## GuiRadioCtrl Member List

This is the complete list of members for `GuiRadioCtrl`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>buttonType</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getText()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>groupNum</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>isField</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember()</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isStateOn()</td>
<td>GuiCheckBoxCtrl</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onControlDragEnter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDoubleClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onMouseDown()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseDragged()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseEnter()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseLeave()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseUp()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove(SimObject object)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onRightClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>performClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td><code>resetState()</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td>setStateOn(bool newState)</td>
<td>GuiCheckBoxCtrl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setText(string text)</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>setTextID(string id)</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>text</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>textID</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>useMouseEvents</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
GuiRolloutCtrl Member List

This is the complete list of members for GuiRolloutCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>autoCollapseSiblings</td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>caption</td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clickCollapse</td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>collapse()</td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlIsChild(control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>defaultHeight</td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>expand()</td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td>expanded</td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getGlobalCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGlobalPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getMinExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getParent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getRoot()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getValue()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hideHeader</code></td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>instantCollapse()</code></td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td><code>instantExpand()</code></td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isActive</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isContainer</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMember(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMouseLocked()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isVisible()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>langTableMod</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>listObjects()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>makeFirstResponder(bool isFirst)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>margin</code></td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td><code>minExtent</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>modal</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>onAction()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onActive(bool state)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onAdd()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>onCollapsed()</td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onExpanded()</td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onHeaderRightClick()</td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>resize</code> (int x, int y, int width, int height)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>save</code> (string fileName, bool selectedOnly=false, string preAppendString='')</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule</code> (float time, string method, string args...)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setActive</code> (bool state=true)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setCanSave</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter</code> (int x, int y)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setExtent</code> (Point2I p)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent</code> (S32 width, S32 height)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setPosition</code> (int x, int y)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setPositionGlobal</code> (int x, int y)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setProfile</code> (GuiControlProfile profile)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sizeToContents()</td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>toggleCollapse()</td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td>toggleExpanded(bool instantly=false)</td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## GuiScriptNotifyCtrl Member List

This is the complete list of members for `GuiScriptNotifyCtrl`, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>isMember</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>isMemberOfClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMouseLocked</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isVisible</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>langTableMod</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>listObjects</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>makeFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>minExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>modal</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>onAction</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onActive</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onAdd</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onChildAdded</code></td>
<td><code>GuiScriptNotifyCtrl</code></td>
</tr>
<tr>
<td><code>onChildAdded</code></td>
<td><code>GuiScriptNotifyCtrl</code></td>
</tr>
<tr>
<td><code>onChildRemoved</code></td>
<td><code>GuiScriptNotifyCtrl</code></td>
</tr>
<tr>
<td><code>onChildRemoved</code></td>
<td><code>GuiScriptNotifyCtrl</code></td>
</tr>
<tr>
<td><code>onChildResized</code></td>
<td><code>GuiScriptNotifyCtrl</code></td>
</tr>
<tr>
<td><code>onChildResized</code></td>
<td><code>GuiScriptNotifyCtrl</code></td>
</tr>
<tr>
<td><code>onControlDragEnter</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onControlDragExit</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>onControlDragged</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder</td>
<td>GuiScriptNotifyCtrl</td>
</tr>
<tr>
<td>onGainFirstResponder(SimObjectId ID)</td>
<td>GuiScriptNotifyCtrl</td>
</tr>
<tr>
<td>GuiControl::onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder</td>
<td>GuiScriptNotifyCtrl</td>
</tr>
<tr>
<td>onLoseFirstResponder(SimObjectId ID)</td>
<td>GuiScriptNotifyCtrl</td>
</tr>
<tr>
<td>GuiControl::onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onParentResized(SimObjectId ID)</td>
<td>GuiScriptNotifyCtrl</td>
</tr>
<tr>
<td>onParentResized</td>
<td>GuiScriptNotifyCtrl</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onResize(SimObjectId ID)</td>
<td>GuiScriptNotifyCtrl</td>
</tr>
<tr>
<td>onSleep(SimObjectId ID)</td>
<td>GuiScriptNotifyCtrl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## GuiScrollCtrl Member List

This is the complete list of members for `GuiScrollCtrl`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>accelerator</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>acceptsAsChild(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>active</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>add(SimObject objects...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>addGuiControl(GuiControl control)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>altCommand</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>anchorBottom</code></td>
<td>GuiContainer</td>
</tr>
<tr>
<td><code>anchorLeft</code></td>
<td>GuiContainer</td>
</tr>
<tr>
<td><code>anchorRight</code></td>
<td>GuiContainer</td>
</tr>
<tr>
<td><code>anchorTop</code></td>
<td>GuiContainer</td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>bringToFront(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>callOnChildren(string method, string args...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>callOnChildrenNoRecurse(string method, string args...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>childMargin</code></td>
<td>GuiScrollCtrl</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clear()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>clearFirstResponder(bool ignored=false)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>computeSizes()</td>
<td>GuiScrollCtrl</td>
</tr>
<tr>
<td>constantThumbHeight</td>
<td>GuiScrollCtrl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>docking</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getGlobalCenter()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGlobalPosition()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getMinExtent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getParent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getRoot()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getScrollPosition()</code></td>
<td><code>GuiScrollCtrl</code></td>
</tr>
<tr>
<td><code>getScrollPositionX()</code></td>
<td><code>GuiScrollCtrl</code></td>
</tr>
<tr>
<td><code>getScrollPositionY()</code></td>
<td><code>GuiScrollCtrl</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getValue</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hScrollBar</td>
<td>GuiScrollCtrl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>lockHorizScroll</td>
<td>GuiScrollCtrl</td>
</tr>
<tr>
<td>lockVertScroll</td>
<td>GuiScrollCtrl</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Field</td>
<td>Type</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>mouseWheelScrollSpeed</td>
<td>GuiScrollCtrl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onScroll()</td>
<td>GuiScrollCtrl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>padding</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(x, y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>pushToBack</td>
<td>(SimObject obj)</td>
</tr>
<tr>
<td>remove</td>
<td>(SimObject objects...)</td>
</tr>
<tr>
<td>reorderChild</td>
<td>(SimObject child1, SimObject child2)</td>
</tr>
<tr>
<td>resize</td>
<td>(int x, int y, int width, int height)</td>
</tr>
<tr>
<td>save</td>
<td>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
</tr>
<tr>
<td>schedule</td>
<td>(float time, string method, string args...)</td>
</tr>
<tr>
<td>scrollToBottom</td>
<td>()</td>
</tr>
<tr>
<td>scrollToObject</td>
<td>(GuiControl control)</td>
</tr>
<tr>
<td>scrollToTop</td>
<td>()</td>
</tr>
<tr>
<td>setActive</td>
<td>(bool state=true)</td>
</tr>
<tr>
<td>setCanSave</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setCenter</td>
<td>(int x, int y)</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>(string name)</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setExtent</td>
<td>(Point2I p)</td>
</tr>
<tr>
<td>setExtent</td>
<td>(S32 width, S32 height)</td>
</tr>
<tr>
<td>setFieldType</td>
<td>(string fieldName, string type)</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>(string fieldName, string value, int index=-1)</td>
</tr>
<tr>
<td>setFilename</td>
<td>(string fileName)</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>()</td>
</tr>
<tr>
<td>setHidden</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setInternalName</td>
<td>(string newInternalName)</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>(bool state=true)</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>(bool state=true)</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setScrollPosition(int x, int y)</td>
<td>GuiScrollCtrl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vScrollBar</td>
<td>GuiScrollCtrl</td>
</tr>
<tr>
<td>willFirstRespond</td>
<td>GuiScrollCtrl</td>
</tr>
</tbody>
</table>
GuiSeparatorCtrl Member List

This is the complete list of members for GuiSeparatorCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>borderMargin</td>
<td>GuiSeparatorCtrl</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>caption</td>
<td>GuiSeparatorCtrl</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td></td>
</tr>
<tr>
<td>clear()</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>extent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControl(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControls(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findObjectByInternalName(string internalName, bool searchChildren=false)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getAspect()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td></td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>invisible</td>
<td>GuiSeparatorCtrl</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>leftMargin</td>
<td>GuiSeparatorCtrl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNameSpace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>type</td>
<td>GuiSeparatorCtrl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
Copyright © GarageGames, LLC. All Rights Reserved.
GuiShapeNameHud Member List

This is the complete list of members for GuiShapeNameHud, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controllsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>distanceFade</code></td>
<td><code>GuiShapeNameHud</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>extent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>fillColor</code></td>
<td><code>GuiShapeNameHud</code></td>
</tr>
<tr>
<td><code>findHitControl(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControls(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findObjectByInternalName(string internalName, bool searchChildren=false)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>frameColor</code></td>
<td><code>GuiShapeNameHud</code></td>
</tr>
<tr>
<td><code>getAspect()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getGlobalCenter()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGlobalPosition()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getMinExtent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getParent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getRoot()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getValue</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isActive()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isContainer</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function/Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method/Field</td>
<td>Type</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType(stringfieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(stringfieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setSuperClassNameNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setValue(string value)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setVisible(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>showFill</code></td>
<td><code>GuiShapeNameHud</code></td>
</tr>
<tr>
<td><code>showFrame</code></td>
<td><code>GuiShapeNameHud</code></td>
</tr>
<tr>
<td><code>sort(string callbackFunction)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>textColor</code></td>
<td><code>GuiShapeNameHud</code></td>
</tr>
<tr>
<td><code>tooltip</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>---------------</td>
<td>------------</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>verticalOffset</td>
<td>GuiShapeNameHud</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
# GuiSliderCtrl Member List

This is the complete list of members for GuiSliderCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsIsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>extent</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>findHitControl(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControls(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findObjectByInternalName(string internalName, bool searchChildren=false)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getAspect()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue()</td>
<td>GuiSliderCtrl</td>
</tr>
<tr>
<td>GuiControl::getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMember</code> (SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>isMemberOfClass</code> (string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod</code> (string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMouseLocked</code> ()</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isThumbBeingDragged</code> ()</td>
<td>GuiSliderCtrl</td>
</tr>
<tr>
<td><code>isVisible</code> ()</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>langTableMod</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>listObjects</code> ()</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>makeFirstResponder</code> (bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>minExtent</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>modal</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>name</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onAction</code> ()</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onActive</code> (bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onAdd</code> ()</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onControlDragEnter</code> (GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onControlDragExit</code> (GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onControlDragged</code> (GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onControlDropped</code> (GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onDialogPop</code> ()</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onDialogPush</code> ()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onMouseDragged()</td>
<td>GuiSliderCtrl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>range</td>
<td>GuiSliderCtrl</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setValue(float pos, bool doCallback=false)</code></td>
<td><code>GuiSliderCtrl</code></td>
</tr>
<tr>
<td><code>GuiControl::setValue(string value)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setVisible(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>snap</code></td>
<td><code>GuiSliderCtrl</code></td>
</tr>
<tr>
<td><code>sort(string callbackFunction)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>ticks</code></td>
<td><code>GuiSliderCtrl</code></td>
</tr>
<tr>
<td><code>tooltip</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>tooltipProfile</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>value</code></td>
<td><code>GuiSliderCtrl</code></td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>----------</td>
<td>------------</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
# GuiSpeedometerHud Member List

This is the complete list of members for GuiSpeedometerHud, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bitmap</td>
<td>GuiBitmapCtrl</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>center</td>
<td>GuiSpeedometerHud</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>color</td>
<td>GuiSpeedometerHud</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>controlsChild</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getGlobalCenter()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGlobalPosition()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getMinExtent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getParent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getRoot()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getValue()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isActive</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isContainer</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>length</td>
<td>GuiSpeedometerHud</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>maxAngle</td>
<td>GuiSpeedometerHud</td>
</tr>
<tr>
<td>maxSpeed</td>
<td>GuiSpeedometerHud</td>
</tr>
<tr>
<td>minAngle</td>
<td>GuiSpeedometerHud</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>onControlDragExit</td>
<td>GuiControl control, Point2I dropPoint</td>
</tr>
<tr>
<td>onControlDragged</td>
<td>GuiControl control, Point2I dropPoint</td>
</tr>
<tr>
<td>onControlDropped</td>
<td>GuiControl control, Point2I dropPoint</td>
</tr>
<tr>
<td>onDialogPop</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded</td>
<td>SimObject object</td>
</tr>
<tr>
<td>onObjectRemoved</td>
<td>SimObject object</td>
</tr>
<tr>
<td>onRemove</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible</td>
<td>bool state</td>
</tr>
<tr>
<td>onWake</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl</td>
<td>int x, int y</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack</td>
<td>SimObject obj</td>
</tr>
<tr>
<td>remove</td>
<td>SimObject objects...</td>
</tr>
<tr>
<td>reorderChild</td>
<td>SimObject child1, SimObject child2</td>
</tr>
<tr>
<td>resize</td>
<td>int x, int y, int width, int height</td>
</tr>
<tr>
<td>save</td>
<td>string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;</td>
</tr>
<tr>
<td>schedule</td>
<td>float time, string method, string</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>args...</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setBitmap(String filename)</td>
<td>GuiBitmapCtrl</td>
</tr>
<tr>
<td>setBitmap(String filename, bool resize)</td>
<td>GuiBitmapCtrl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(int x, int y)</td>
<td>GuiBitmapCtrl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>GuiControl::setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tail</td>
<td>GuiSpeedometerHud</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
<tr>
<td>width</td>
<td>GuiSpeedometerHud</td>
</tr>
<tr>
<td>wrap</td>
<td>GuiBitmapCtrl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
GuiSplitContainer Member List

This is the complete list of members for `GuiSplitContainer`, including all inherited members.

<table>
<thead>
<tr>
<th>Method/Property</th>
<th>Class/Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>anchorBottom</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorLeft</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorRight</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorTop</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Command</td>
<td>Return Type</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>controlsChild(GuiControl control)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>docking</code></td>
<td>GuiContainer</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>extent</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findHitControl(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findHitControls(int x, int y, int width, int height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findObjectByInternalName(string internalName, bool searchChildren=false)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>fixedPanel</code></td>
<td>GuiSplitContainer</td>
</tr>
<tr>
<td><code>fixedSize</code></td>
<td>GuiSplitContainer</td>
</tr>
<tr>
<td><code>getAspect()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td><code>onControlDragged</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onControlDropped</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onDialogPop</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onDialogPush</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onGainFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onLoseFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onObjectAdded</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onObjectRemoved</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onRemove</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onSleep</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onVisible</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onWake</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>orientation</code></td>
<td>GuiSplitContainer</td>
</tr>
<tr>
<td><code>padding</code></td>
<td>GuiContainer</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>pointInControl</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>position</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>profile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>pushToBack</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>remove</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>reorderChild</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>resize</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>save</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>schedule</code> (float time, string method, string args...)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setActive</code> (bool state=true)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setCanSave</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter</code> (int x, int y)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setExtent</code> (Point2I p)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent</code> (S32 width, S32 height)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setPosition</code> (int x, int y)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setPositionGlobal</code> (int x, int y)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setProfile</code> (GuiControlProfile profile)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setValue</code> (string value)</td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>splitPoint</td>
<td>GuiSplitContainer</td>
</tr>
<tr>
<td>splitterSize</td>
<td>GuiSplitContainer</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
## GuiStackControl Member List

This is the complete list of members for **GuiStackControl**, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>changeChildPosition</td>
<td>GuiStackControl</td>
</tr>
<tr>
<td>changeChildSizeToFit</td>
<td>GuiStackControl</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dynamicNonStackExtent</code></td>
<td><code>GuiStackControl</code></td>
</tr>
<tr>
<td><code>dynamicPos</code></td>
<td><code>GuiStackControl</code></td>
</tr>
<tr>
<td><code>dynamicSize</code></td>
<td><code>GuiStackControl</code></td>
</tr>
<tr>
<td><code>extent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControl(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControls(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findObjectByInternalName(string internalName, bool searchChildren=false)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>freeze(bool freeze)</code></td>
<td><code>GuiStackControl</code></td>
</tr>
<tr>
<td><code>getAspect()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectId(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>horizStacking</td>
<td>GuiStackControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>isChildOfGroup(group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isFrozen()</td>
<td>GuiStackControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(control, dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(control, dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>dropPoint()</td>
<td></td>
</tr>
<tr>
<td><strong>onControlDragged</strong>(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>onControlDropped</strong>(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>onObjectAdded</strong>((SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td><strong>onObjectRemoved</strong>((SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>padding</td>
<td>GuiStackControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>pointInControl</strong>(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>pushToBack</strong>(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td><strong>remove</strong>((SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td><strong>reorderChild</strong>((SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td><strong>resize</strong>(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><strong>save</strong>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNameNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>Property</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>stackingType</td>
<td>GuiStackControl</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>updateStack()</td>
<td>GuiStackControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertStacking</td>
<td>GuiStackControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
GuiSwatchButtonCtrl Member List

This is the complete list of members for **GuiSwatchButtonCtrl**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>buttonType</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>color</td>
<td>GuiSwatchButtonCtrl</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassname()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getGlobalCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGlobalPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getMinExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getParent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getRoot()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getText()</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>getValue</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>groupNum</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isActive</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isContainer</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
onControlDropped (GuiControl control, Point2I dropPoint)

onDialogPop()

onDialogPush()

onDoubleClick()

onGainFirstResponder()

onLoseFirstResponder()

onMouseDown()

onMouseDragged()

onMouseEnter()

onMouseLeave()

onMouseUp()

onObjectAdded (SimObject object)

onObjectRemoved (SimObject object)

onRemove (SimObject object)

onRightClick()

onSleep()

onVisible (bool state)

onWake()

parentGroup

performClick()

persistentId

pointInControl (int x, int y)

position

profile

pushToBack (SimObject obj)

remove (SimObject objects...)

reorderChild (SimObject child1, SimObject child2)
<table>
<thead>
<tr>
<th>Method</th>
<th>Class/Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>resetState()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setColor(string newColor)</td>
<td>GuiSwatchButtonCtrl</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setStateOn(bool isOn=true)</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setText(string text)</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>setTextID(string id)</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>setValue(string value)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setVisible(bool state=true)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>sort(string callbackFunction)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>text</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>textID</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>tooltip</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>tooltipProfile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>useMouseEvents</code></td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td><code>variable</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>vertSizing</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>visible</code></td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
GuiTabBookCtrl Member List

This is the complete list of members for GuiTabBookCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>addPage(string title=&quot;&quot;)</td>
<td>GuiTabBookCtrl</td>
</tr>
<tr>
<td>allowReorder</td>
<td>GuiTabBookCtrl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>anchorBottom</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorLeft</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorRight</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorTop</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>defaultPage</td>
<td>GuiTabBookCtrl</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>docking</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>frontTabPadding</td>
<td>GuiTabBookCtrl</td>
</tr>
<tr>
<td>get Aspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>.getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSelectedPage()</td>
<td>GuiTabBookCtrl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isActive</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isContainer</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMember(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMouseLocked()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isVisible()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>langTableMod</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>listObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>makeFirstResponder(bool isFirst)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>margin</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>minExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>minTabWidth</code></td>
<td><code>GuiTabBookCtrl</code></td>
</tr>
<tr>
<td><code>modal</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>onAction()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>onActive(bool state)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Method/Function</td>
<td>Type/Description</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onTabRightClick(String text, int index)</td>
<td>GuiTabBookCtrl</td>
</tr>
<tr>
<td>onTabSelected(String text, int index)</td>
<td>GuiTabBookCtrl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>padding</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>reorderChild</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>selectedPage</td>
<td>GuiTabBookCtrl</td>
</tr>
<tr>
<td>selectPage</td>
<td>GuiTabBookCtrl</td>
</tr>
<tr>
<td>setActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class/Type</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tabHeight</td>
<td>GuiTabBookCtrl</td>
</tr>
<tr>
<td>tabMargin</td>
<td>GuiTabBookCtrl</td>
</tr>
<tr>
<td>tabPosition</td>
<td>GuiTabBookCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## GuiTabPageCtrl Member List

This is the complete list of members for **GuiTabPageCtrl**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>anchorBottom</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorLeft</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorRight</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorTop</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>docking</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>fitBook</td>
<td>GuiTabPageCtrl</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectId(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>maxLength</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>dropPoint()</td>
<td></td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>padding</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>select()</td>
<td>GuiTabPageCtrl</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setText(string text)</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>setTextID(string textID)</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>text</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>textID</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
**GuiTextCtrl Member List**

This is the complete list of members for `GuiTextCtrl`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>accelerator</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>acceptsAsChild(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>active</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>add(SimObject objects...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>addGuiControl(GuiControl control)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>altCommand</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>anchorBottom</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>anchorLeft</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>anchorRight</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>anchorTop</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>bringToFront(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>callOnChildren(string method, string args...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>callOnChildrenNoRecurse(string method, string args...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clear()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>clearFirstResponder(bool ignored=false)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>docking</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>getFieldType</code> (string fieldName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue</code> (string fieldName, int index=-1)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getFullCount</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getGlobalCenter</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGlobalPosition</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getMinExtent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getObjectIndex</code> (SimObject obj)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getParent</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getPosition</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getRandom</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getRoot</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getValue</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>horizSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>hovertime</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isActive</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isAwake()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>isChildOfGroup</code> (SimGroup group)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isContainer</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>-langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimSet</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>maxLength</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td><code>onControlDragged</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onControlDropped</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onDialogPop</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onDialogPush</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onGainFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onLoseFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onObjectAdded</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onObjectRemoved</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onRemove</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onSleep</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onVisible</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onWake</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>padding</code></td>
<td>GuiContainer</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>pointInControl</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>position</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>profile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>pushToBack</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>remove</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>reorderChild</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>resize</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>save</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setActive</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setText(string text)</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>setTextID(string textID)</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>term</td>
<td>type</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>text</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>textID</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
**GuiTextEditCtrl Member List**

This is the complete list of members for *GuiTextEditCtrl*, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>anchorBottom</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorLeft</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorRight</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorTop</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clearSelectedText()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>deniedSound</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>docking</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>escapeCommand</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>forceValidateText()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getCursorPos()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTypeField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getText()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>getValue()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>historySize</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>Function/Method</td>
<td>Class/Type</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAllTextSelected()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>maxLength</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onReturn()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onTabComplete(string val)</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>onValidate()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>padding</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>password</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>passwordMask</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>selectAllText</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>setActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCursorPos</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setText(string text)</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>setTextID(string textID)</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sinkAllKeyEvents</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tabComplete</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>text</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>textID</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>validate</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
# GuiTextEditSliderBitmapCtrl Member List

This is the complete list of members for `GuiTextEditSliderBitmapCtrl`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>accelerator</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>acceptsAsChild(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>active</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>add(SimObject objects...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>addGuiControl(GuiControl control)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>altCommand</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>anchorBottom</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>anchorLeft</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>anchorRight</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>anchorTop</code></td>
<td><code>GuiContainer</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>bitmap</code></td>
<td><code>GuiTextEditSliderBitmapCtrl</code></td>
</tr>
<tr>
<td><code>bringToFront(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>callOnChildren(string method, string args...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>callOnChildrenNoRecurse(string method, string args...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clear()</code></td>
<td><code>SimSet</code></td>
</tr>
</tbody>
</table>
clearFirstResponder(bool ignored=false)

clearSelectedText()

close()

clone() SimObject

command GuiControl

controlsChild(GuiControl control) GuiControl

depthClone()

delete() SimObject

deleteAllObjects() SimSet

deniedSound GuiTextEditCtrl

docking GuiContainer

dump(bool detailed=false) SimObject

dumpClassHierarchy() SimObject

dumpGroupHierarchy() SimObject

dumpMethods() SimObject

escapeCommand GuiTextEditCtrl

extent GuiControl

findHitControl(int x, int y) GuiControl

findHitControls(int x, int y, int width, int height) GuiControl

findObjectByInternalName(string internalName, bool searchChildren=false) SimSet

focusOnMouseWheel GuiTextEditSliderBitmapCtrl

forceValidateText() GuiTextEditCtrl

format GuiTextEditSliderBitmapCtrl

getAspect() GuiControl

getCanSave() SimObject

getCenter() GuiControl
<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNameNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getCursorPos()</code></td>
<td><code>GuiTextEditCtrl</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getGlobalCenter()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGlobalPosition()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getMinExtent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getParent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getText()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>historySize</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>increment</td>
<td>GuiTextEditSliderBitmapCtrl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAllTextSelected()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>maxLength</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onReturn()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>Method</td>
<td>Class/Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onTabComplete(string val)</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>onValidate()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>padding</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>password</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>passwordMask</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>range</td>
<td>GuiTextEditSliderBitmapCtrl</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>selectAllText()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>setCursorPos(int position)</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNameamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setText(string text)</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>setTextID(string textID)</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>sinkAllKeyEvents</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>sort (string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tabComplete</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>text</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>textID</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>validate</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
This is the complete list of members for **GuiTextEditSliderCtrl**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>anchorBottom</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorLeft</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorRight</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorTop</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clearSelectedText()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>deniedSound</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>docking</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>escapeCommand</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>extent</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>focusOnMouseWheel</td>
<td>GuiTextEditSliderCtrl</td>
</tr>
<tr>
<td>forceValidateText()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>format</td>
<td>GuiTextEditSliderCtrl</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>getCursorPos()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getText()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>historySize</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>horvertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>increment</td>
<td>GuiTextEditSliderCtrl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAllTextSelected()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Member</td>
<td>Type</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>maxLength</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>padding</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>password</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>passwordMask</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>range</td>
<td>GuiTextEditSliderCtrl</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>selectAllText()</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNameNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCursorPos(int position)</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setText</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>setTextID</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>setValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sinkAllKeyEvents</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>sort</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tabComplete</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>text</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>textID</td>
<td>GuiTextCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
</tr>
<tr>
<td>------------</td>
<td>----------------</td>
</tr>
<tr>
<td>validate</td>
<td>GuiTextEditCtrl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
# GuiTextListCtrl Member List

This is the complete list of members for GuiTextListCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>addRow(int id=0, string text=&quot;&quot;, int index=-1)</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>clear()</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>clearSelection()</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>clipColumnText</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>columns</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>findTextIndex(string needle)</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>fitParentWidth</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>getId(int index)</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>getRowNumById(int id)</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>getRowText(int index)</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>getRowTextById(int id)</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>getSelectedId()</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>getSelectedRow()</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>isRowActive(int rowNum)</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>onDeleteKey(string id)</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>onSelect(string cellid, string text)</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>removeRow(int index)</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>removeRowById(int id)</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>rowCount()</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>scrollVisible(int rowNum)</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>setRowActive(int rowNum, bool active)</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>setRowById(int id, string text)</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>setSelectedById(int id)</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>setSelectedRow(int rowNum)</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td><code>sort(int columnId, bool increasing=true)</code></td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td><code>sortNumerical(int columnID, bool increasing=true)</code></td>
<td>GuiTextListCtrl</td>
</tr>
</tbody>
</table>
## GuiTheoraCtrl Member List

This is the complete list of members for `GuiTheoraCtrl`, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>backgroundColor</td>
<td>GuiTheoraCtrl</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getCurrentTime()</td>
<td>GuiTheoraCtrl</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFileType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>isFirstFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isPlaybackDone()</td>
<td>GuiTheoraCtrl</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>matchVideoSize</td>
<td>GuiTheoraCtrl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pause()</td>
<td>GuiTheoraCtrl</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>play()</td>
<td>GuiTheoraCtrl</td>
</tr>
<tr>
<td>playOnWake</td>
<td>GuiTheoraCtrl</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>renderDebugInfo</td>
<td>GuiTheoraCtrl</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNameSpace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFile(string filename)</td>
<td>GuiTheoraCtrl</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNameSpace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>stop()</td>
<td>GuiTheoraCtrl</td>
</tr>
<tr>
<td>stopOnSleep</td>
<td>GuiTheoraCtrl</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>theoraFile</td>
<td>GuiTheoraCtrl</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>transcoder</td>
<td>GuiTheoraCtrl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
GuiTickCtrl Member List

This is the complete list of members for GuiTickCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurs (string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsIsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>extent</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>findHitControl(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findHitControls(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>findObjectByInternalName(string internalName, bool searchChildren=false)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getAspect()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>----------</td>
<td>------</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>onObjectAdded</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onObjectRemoved</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onRemove()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onSleep()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onVisible(bool state)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>onWake()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>pointInControl(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>position</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>profile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>pushToBack(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>remove(SimObject objects...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>reorderChild(SimObject child1, SimObject child2)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimControl</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPosition(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setPositionGlobal(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setProcessTicks(bool tick)</td>
<td>GuiTickCtrl</td>
</tr>
<tr>
<td>setProfile(GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
# GuiToggleButtonCtrl Member List

This is the complete list of members for `GuiToggleButtonCtrl`, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>buttonType</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearFirstResponder(bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectId(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getText()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>groupNum</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Returns</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDoubleClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onMouseDown()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseDragged()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseEnter()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseLeave()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onMouseUp()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onRightClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>performClick()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resetState()</td>
<td>GuiButtonBaseCtrl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>resize(int x, int y, int width, int height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setActive(bool state=true)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCenter(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setExtent(Point2I p)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setExtent(S32 width, S32 height)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setPosition(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setPositionGlobal(int x, int y)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setProfile(GuiControlProfile profile)</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setStateOn(bool isOn=true)</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setText</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>setTextID</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>setValue</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>setVisible</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>sort</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>text</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>textID</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>tooltip</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>tooltipProfile</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>useMouseEvents</code></td>
<td><code>GuiButtonBaseCtrl</code></td>
</tr>
<tr>
<td><code>variable</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>vertSizing</code></td>
<td><code>GuiControl</code></td>
</tr>
<tr>
<td><code>visible</code></td>
<td><code>GuiControl</code></td>
</tr>
</tbody>
</table>
# GuiTreeViewCtrl Member List

This is the complete list of members for `GuiTreeViewCtrl`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>addChildSelectionByValue</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>addSelection(int id, bool isLastSelection=true)</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>buildIconTable</code></td>
<td><code>builds an icon table</code></td>
</tr>
<tr>
<td><code>buildVisibleTree</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>cancelRename</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>canRenameObject(SimObject object)</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>canRenameObjects</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>clear()</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>clearAllOnSingleSelection</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>clearFilterText()</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>clearSelection()</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>compareToObjectID</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>deleteObjectAllowed</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>deleteSelection()</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>destroyTreeOnSleep</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>dragToItemAllowed</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>editItem(TreeItemId item, string newText, string newValue)</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>expandItem(TreeItemId item, bool expand=true)</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>findChildItemByName(int parentId, string childName)</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>findItemByName(string text)</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td><code>findItemByObjectId</code></td>
<td><code>find item by object id and returns the mId</code></td>
</tr>
<tr>
<td><code>findItemByValue(string value)</code></td>
<td><code>GuiTreeviewCtrl</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>fullRowSelect</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>getChild(TreeItemId item)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>getFilterText()</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>getFirstRootItem</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>getItemCount</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>getItemText(TreeItemId item)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>getItemValue(TreeItemId item)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>getNextSibling(TreeItemId item)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>getParent(TreeItemId item)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>getPrevSibling(TreeItemId item)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>getSelectedltem(int index=0)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>getSelectedltemList</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>getSelectedltemsCount</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>getSelectedObject(int index=0)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>getSelectedObjectList</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>getTextToRoot(TreeItemId item, Delimiter=none)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>handleRenameObject(string newName, SimObject object)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>hideSelection(bool state=true)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>insertItem(int parentId, string text, string value=&quot;&quot;, string icon=&quot;&quot;, int normalImage=0, int expandedImage=0)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>isItemSelected(int id)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>isParentltem(int id)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>isValidDragTarget(int id, string value)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>itemHeight</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>lockSelection(bool lock=true)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>markItem(TreeItemId item, bool mark=true)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>Event Method</td>
<td>Argument(s)</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td><code>mouseDragging</code></td>
<td></td>
</tr>
<tr>
<td><code>moveItemDown(TreeItemId item)</code></td>
<td></td>
</tr>
<tr>
<td><code>moveItemUp(TreeItemId item)</code></td>
<td></td>
</tr>
<tr>
<td><code>multipleSelections</code></td>
<td></td>
</tr>
<tr>
<td><code>onAddGroupSelected(SimGroup group)</code></td>
<td></td>
</tr>
<tr>
<td><code>onAddMultipleSelectionBegin()</code></td>
<td></td>
</tr>
<tr>
<td><code>onAddMultipleSelectionEnd()</code></td>
<td></td>
</tr>
<tr>
<td><code>onAddSelection</code></td>
<td>(int itemOrObjectId, bool isLastSelection)</td>
</tr>
<tr>
<td><code>onBeginReparenting()</code></td>
<td></td>
</tr>
<tr>
<td><code>onClearSelection()</code></td>
<td></td>
</tr>
<tr>
<td><code>onDefineIcons()</code></td>
<td></td>
</tr>
<tr>
<td><code>onDeleteObject</code></td>
<td>(SimObject object)</td>
</tr>
<tr>
<td><code>onDeleteSelection()</code></td>
<td></td>
</tr>
<tr>
<td><code>onDragDropped()</code></td>
<td></td>
</tr>
<tr>
<td><code>onEndReparenting()</code></td>
<td></td>
</tr>
<tr>
<td><code>onInspect</code></td>
<td>(int itemOrObjectId)</td>
</tr>
<tr>
<td><code>onKeyDown</code></td>
<td>(int modifier, int keyCode)</td>
</tr>
<tr>
<td><code>onMouseDragged()</code></td>
<td></td>
</tr>
<tr>
<td><code>onMouseUp</code></td>
<td>(int hitItemId, int mouseClickCount)</td>
</tr>
<tr>
<td><code>onObjectDeleteCompleted()</code></td>
<td></td>
</tr>
<tr>
<td><code>onRemoveSelection</code></td>
<td>(int itemOrObjectId)</td>
</tr>
<tr>
<td><code>onRenameValidate</code></td>
<td></td>
</tr>
<tr>
<td><code>onReparent</code></td>
<td>(int itemOrObjectId, int oldParentItemOrObjectId, int newParentItemOrObjectId)</td>
</tr>
<tr>
<td><code>onRightMouseDow</code></td>
<td>(int itemId, Point2I mousePos, SimObject object)</td>
</tr>
<tr>
<td><code>onRightMouseUp</code></td>
<td>(int itemId, Point2I mousePos,)</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SimObject object)</td>
<td></td>
</tr>
<tr>
<td><strong>onSelect</strong> (int itemOrObjectId)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>onUnselect</strong> (int itemOrObjectId)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>open</strong> (SimSet obj, bool okToEdit=true)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>removeAllChildren</strong></td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>removeChildSelectionByValue</strong></td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>removeItem</strong> (TreeItemId item)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>removeSelection</strong> (deselects an item)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>renameInternal</strong></td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>scrollVisible</strong> (TreeItemId item)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>scrollVisibleByObjectId</strong></td>
<td>GuiTreeCtrl (show item by object id. returns true if successful.)</td>
</tr>
<tr>
<td><strong>selectItem</strong> (TreeItemId item, bool select=true)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>setDebug</strong> (bool value=true)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>setFilterText</strong> (string pattern)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>setItemImages</strong> (int id, int normalImage, int expandedImage)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>setItemTooltip</strong> (int id, string text)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>showClassNameForUnnamedObjects</strong></td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>showClassNames</strong></td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>showInternalNames</strong></td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>showItemRenameCtrl</strong> (TreeItemId id)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>showObjectIds</strong></td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>showObjectNames</strong></td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>showRoot</strong></td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>sort</strong> (int parent, bool traverseHierarchy=false, bool parentsFirst=false, bool caseSensitive=true)</td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>tabSize</strong></td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td><strong>textOffset</strong></td>
<td>GuiTreeCtrl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>toggleHideSelection()</td>
<td>GuiTreeViewCtrl</td>
</tr>
<tr>
<td>toggleLockSelection()</td>
<td>GuiTreeViewCtrl</td>
</tr>
<tr>
<td>tooltipOnWidthOnly</td>
<td>GuiTreeViewCtrl</td>
</tr>
<tr>
<td>useInspectorTooltips</td>
<td>GuiTreeViewCtrl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## GuiTSCtrl Member List

This is the complete list of members for GuiTSCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>anchorBottom</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorLeft</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorRight</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorTop</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>calculateViewDistance(float radius)</td>
<td>GuiTSCtrl</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>cameraZRot</td>
<td>GuiTSCtrl</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>Method Description</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>clearFirstResponder</code> (bool ignored=false)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>command</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>controlsChild(GuiControl control)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>docking</code></td>
<td>GuiContainer</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>extent</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findHitControl(int x, int y)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findHitControls(int x, int y, int width, int height)</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>findObjectByInternalName(string internalName, bool searchChildren=false)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>forceFOV</code></td>
<td>GuiTSCtrl</td>
</tr>
<tr>
<td><code>getAsp</code>ct()`</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td>getExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGlobalCenter()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGlobalPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getMinExtent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getParent()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getPosition()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getWorldToScreenScale()</td>
<td>GuiTSCtrl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>dropPoint)</td>
<td></td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>padding</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>project(Point3F worldPosition)</td>
<td>GuiTSCtrl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reflectPriority</td>
<td>GuiTSCtrl</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>resize</code> (int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>save</code> (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule</code> (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setActive</code> (bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setCanSave</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCenter</code> (int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setExtent</code> (Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setExtent</code> (S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setPosition</code> (int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setPositionGlobal</code> (int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setProfile</code> (GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>setValue(string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setVisible(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tooltip</td>
<td>GuiControl</td>
</tr>
<tr>
<td>tooltipProfile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>unproject(Point3F screenPosition)</td>
<td>GuiTSCtrl</td>
</tr>
<tr>
<td>variable</td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>
GuiWindowCtrl Member List

This is the complete list of members for GuiWindowCtrl, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerator</td>
<td>GuiControl</td>
</tr>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>active</td>
<td>GuiControl</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addGuiControl(GuiControl control)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>altCommand</td>
<td>GuiControl</td>
</tr>
<tr>
<td>anchorBottom</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorLeft</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorRight</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>anchorTop</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>attach(GuiWindowCtrl bottomWindow, GuiWindowCtrl topWindow)</td>
<td>GuiWindowCtrl [static]</td>
</tr>
<tr>
<td>attachTo(GuiWindowCtrl window)</td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canClose</td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td>canCollapse</td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td>canMaximize</td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td>canMinimize</td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>canMove</td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearFirstResponder</td>
<td>GuiControl</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>closeCommand</td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td>command</td>
<td>GuiControl</td>
</tr>
<tr>
<td>controlsChild</td>
<td>GuiControl</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>docking</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>edgeSnap</td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td>extent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findHitControls(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getAspect()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getGlobalCenter()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGlobalPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getMinExtent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getParent()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRoot()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>horizSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>hovertime</td>
<td>GuiControl</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isAwake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isContainer</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMouseLocked()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isVisible()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>langTableMod</td>
<td>GuiControl</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>makeFirstResponder(bool isFirst)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>margin</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>minExtent</td>
<td>GuiControl</td>
</tr>
<tr>
<td>modal</td>
<td>GuiControl</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAction()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onActive(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onAdd()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onClose()</td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td>onCollapse()</td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td>onControlDragEnter(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragExit(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDragged(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onControlDropped(GuiControl control, Point2I dropPoint)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPop()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onDialogPush()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onGainFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onLoseFirstResponder()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onMaximize()</td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td>onMinimize()</td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onRemove()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onRestore()</td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td>onSleep()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>onVisible(bool state)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>onWake()</td>
<td>GuiControl</td>
</tr>
<tr>
<td>padding</td>
<td>GuiContainer</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pointInControl(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>position</td>
<td>GuiControl</td>
</tr>
<tr>
<td>profile</td>
<td>GuiControl</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resize(int x, int y, int width, int height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>resizeHeight</td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td>resizeWidth</td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>selectWindow()</td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td>setActive(bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCenter(int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCollapseGroup(bool state)</td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setExtent(Point2I p)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setExtent(S32 width, S32 height)</td>
<td>GuiControl</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string }</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Object Type</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFirstResponder</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setFirstResponder()</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setPosition</code> (int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setPositionGlobal</code> (int x, int y)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setProfile</code> (GuiControlProfile profile)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setSuperClassNameSpace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setValue</code> (string value)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>setVisible</code> (bool state=true)</td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>sort</code> (string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>text</code></td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td><code>toggleCollapseGroup()</code></td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td><code>tooltip</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>tooltipProfile</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td><code>variable</code></td>
<td>GuiControl</td>
</tr>
<tr>
<td>vertSizing</td>
<td>GuiControl</td>
</tr>
<tr>
<td>------------</td>
<td>------------</td>
</tr>
<tr>
<td>visible</td>
<td>GuiControl</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# HoverVehicle Member List

This is the complete list of members for `HoverVehicle`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>applyDamage(float amount)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>applyImpulse(Point3F pos, Point3F vec)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>GameBase::applyImpulse(Point3F pos, VectorF vel)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>applyRadialImpulse(Point3F origin, float radius, float magnitude)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>applyRepair(float amount)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>boundingBox</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canCloak()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>changeMaterial(string mapTo, Material oldMat, Material newMat)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dataBlock</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>destroyThread(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>disableMove</code></td>
<td>Vehicle</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMeshVisibility()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getAIRepairPoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getControllingClient()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getControllingObject()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageFlash()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamagePercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageState()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDefaultCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEnergyLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEnergyPercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getEyePoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeTransform()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td><code>getEyeVector()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getImageAltTrigger(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageAmmo(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageGenericTrigger(int slot, int trigger)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageLoaded(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageScriptAnimPrefix(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageSkinTag(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageState(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageTarget(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageTrigger(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getLookAtPoint(float distance=2000, int typeMask=0xFFFFFFFF)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code> getMaxDamage()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getModelFile()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMountedImage(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountSlot(ShapeBaseImageData)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzlePoint(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzleVector(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPendingImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRechargeRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRepairRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getShapeName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSkinName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSlotTransform(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTargetCount()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTargetName(int index)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getVelocity()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getWhiteOut()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hasImageState(int slot, string state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isAIControlled</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isCloaked()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDestroyed()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDisabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEnabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isHidden()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageFiring(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageMounted(ShapeBaseImageData image)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>HoverVehicle</code></td>
</tr>
<tr>
<td><code>isRenderEnabled()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>HoverVehicle</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountImage(ShapeBaseImageData image, int slot, bool loaded=true, string skinTag=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pauseThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAudio(int slot, SFXTrack track)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>playThread(int slot, string name=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setAllMeshesHidden(bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>setCanSave</td>
<td>Set if object can be saved.</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>Set object's class namespace.</td>
</tr>
<tr>
<td>setCloaked</td>
<td>Set if object is cloaked.</td>
</tr>
<tr>
<td>setControl</td>
<td>Set if object is controlled.</td>
</tr>
<tr>
<td>setDamageFlash</td>
<td>Set object's damage flash level.</td>
</tr>
<tr>
<td>setDamageLevel</td>
<td>Set object's damage level.</td>
</tr>
<tr>
<td>setDamageState</td>
<td>Set object's damage state.</td>
</tr>
<tr>
<td>setDamageVector</td>
<td>Set object's damage vector.</td>
</tr>
<tr>
<td>setDataBlock</td>
<td>Set object's data block.</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>Set if object is editor-only.</td>
</tr>
<tr>
<td>setEnergyLevel</td>
<td>Set object's energy level.</td>
</tr>
<tr>
<td>setFieldType</td>
<td>Set object's field type.</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>Set object's field value.</td>
</tr>
<tr>
<td>setFilename</td>
<td>Set object's filename.</td>
</tr>
<tr>
<td>setHidden</td>
<td>Set if object is hidden.</td>
</tr>
<tr>
<td>setImageAltTrigger</td>
<td>Set object's alternate trigger state.</td>
</tr>
<tr>
<td>setImageAmmo</td>
<td>Set object's ammo trigger state.</td>
</tr>
<tr>
<td>setImageGenericTrigger</td>
<td>Set object's generic trigger state.</td>
</tr>
<tr>
<td>setImageLoaded</td>
<td>Set object's loaded state.</td>
</tr>
<tr>
<td>setImageScriptAnimPrefix</td>
<td>Set object's script animation prefix.</td>
</tr>
<tr>
<td>setImageTarget</td>
<td>Set object's target state.</td>
</tr>
<tr>
<td>setImageTrigger</td>
<td>Set object's trigger state.</td>
</tr>
<tr>
<td>setInternalName</td>
<td>Set object's internal name.</td>
</tr>
<tr>
<td>setInvincibleMode</td>
<td>Set object's invincible mode.</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>Set if object is expanded.</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>Set if object is selected.</td>
</tr>
</tbody>
</table>

All methods are part of the SimObject, ShapeBase, and GameBase classes, as indicated by their return types.
<table>
<thead>
<tr>
<th>Function</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setMeshHidden(string name, bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setRechargeRate(float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setRepairRate(float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setShapeName(string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setSkinName(string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setThreadDir(int slot, bool fwd)</td>
<td>ShapeObject</td>
</tr>
<tr>
<td>setThreadPosition(int slot, float pos)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setThreadTimeScale(int slot, float scale)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setTransform(TransformF tfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setVelocity(Point3F vel)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setWhiteOut(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>skin</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>startFade(int time, int delay, bool fadeOut)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>stopAudio(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>stopThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>validateCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>workingQueryBoxSizeMultiplier</td>
<td>Vehicle</td>
</tr>
<tr>
<td>workingQueryBoxStaleThreshold</td>
<td>Vehicle</td>
</tr>
</tbody>
</table>
# HoverVehicleData Member List

This is the complete list of members for `HoverVehicleData`, including all inherited members.

<table>
<thead>
<tr>
<th>Method/Property</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>bodyFriction</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>bodyRestitution</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>brakingActivationSpeed</code></td>
<td><code>HoverVehicleData</code></td>
</tr>
<tr>
<td><code>brakingForce</code></td>
<td><code>HoverVehicleData</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>cameraDecay</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>cameraDefaultFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraLag</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>cameraMaxDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMaxFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraOffset</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>cameraRoll</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>category</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>checkDeployPos(TransformF txfm)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>collDamageMultiplier</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>collDamageThresholdVel</td>
<td>VehicleData</td>
</tr>
<tr>
<td>collisionTol</td>
<td>VehicleData</td>
</tr>
<tr>
<td>computeCRC</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>contactTol</td>
<td>VehicleData</td>
</tr>
<tr>
<td>cubeReflectorDesc</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>damageEmitter</td>
<td>VehicleData</td>
</tr>
<tr>
<td>damageEmitterOffset</td>
<td>VehicleData</td>
</tr>
<tr>
<td>damageLevelTolerance</td>
<td>VehicleData</td>
</tr>
<tr>
<td>Debris</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>debrisShapeName</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>density</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>destroyedLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>disabledLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>drag</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>dragForce</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dustEmitter</td>
<td>VehicleData</td>
</tr>
<tr>
<td>dustHeight</td>
<td>VehicleData</td>
</tr>
<tr>
<td>dustTrailEmitter</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>dustTrailFreqMod</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>dustTrailOffset</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>engineSound</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>exitingWater</td>
<td>VehicleData</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>exitSplashSoundVelocity</td>
<td>VehicleData</td>
</tr>
<tr>
<td>Explosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>firstPersonOnly</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>floatingGravMag</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>floatingThrustFactor</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>floatSound</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>forwardJetEmitter</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeployTransform(Point3F pos, Point3F normal)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>gyroDrag</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>hardImpactSound</td>
<td>VehicleData</td>
</tr>
<tr>
<td>Function/Property</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>hardImpactSpeed</td>
<td>VehicleData</td>
</tr>
<tr>
<td>hardSplashSoundVelocity</td>
<td>VehicleData</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>impactWaterEasy</td>
<td>VehicleData</td>
</tr>
<tr>
<td>impactWaterHard</td>
<td>VehicleData</td>
</tr>
<tr>
<td>impactWaterMedium</td>
<td>VehicleData</td>
</tr>
<tr>
<td>inheritEnergyFromMount</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>integration</td>
<td>VehicleData</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInvincible</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>jetEnergyDrain</td>
<td>VehicleData</td>
</tr>
<tr>
<td>jetForce</td>
<td>VehicleData</td>
</tr>
<tr>
<td>jetSound</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mainThrustForce</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>mass</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>massBox</td>
<td>VehicleData</td>
</tr>
<tr>
<td>massCenter</td>
<td>VehicleData</td>
</tr>
<tr>
<td>maxDamage</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>maxDrag</td>
<td>VehicleData</td>
</tr>
<tr>
<td>maxEnergy</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>maxSteeringAngle</td>
<td>VehicleData</td>
</tr>
<tr>
<td>mediumSplashSoundVelocity</td>
<td>VehicleData</td>
</tr>
<tr>
<td>minDrag</td>
<td>VehicleData</td>
</tr>
<tr>
<td>minImpactSpeed</td>
<td>VehicleData</td>
</tr>
<tr>
<td>minJetEnergy</td>
<td>VehicleData</td>
</tr>
<tr>
<td>minRollSpeed</td>
<td>VehicleData</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>normalForce</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>numDmgEmitterAreas</td>
<td>VehicleData</td>
</tr>
<tr>
<td>observeThroughObject</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onAdd(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onCollision(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onDamage(ShapeBase obj, float delta)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onDestroyed(ShapeBase obj, string lastState)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onDisabled(ShapeBase obj, string lastState)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onEnabled(ShapeBase obj, string lastState)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onEndSequence(ShapeBase obj, int slot)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onEnterLiquid(Vehicle obj, float coverage, string type)</td>
<td>VehicleData</td>
</tr>
<tr>
<td>onForceUncloak(ShapeBase obj, string reason)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onImpact(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onLeaveLiquid(Vehicle obj, string type)</td>
<td>VehicleData</td>
</tr>
<tr>
<td>onMount(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>onNewDataBlock</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onRemove</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onTrigger</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onUnmount</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>pitchForce</code></td>
<td><code>HoverVehicleData</code></td>
</tr>
<tr>
<td><code>reloadOnLocalClient</code></td>
<td><code>SimDataBlock</code></td>
</tr>
<tr>
<td><code>renderWhenDestroyed</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>repairRate</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>restorativeForce</code></td>
<td><code>HoverVehicleData</code></td>
</tr>
<tr>
<td><code>reverseThrustForce</code></td>
<td><code>HoverVehicleData</code></td>
</tr>
<tr>
<td><code>rollForce</code></td>
<td><code>HoverVehicleData</code></td>
</tr>
<tr>
<td><code>save</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNameSpace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code> setHidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>shadowEnable</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowMaxVisibleDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowProjectionDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSize</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSphereAdjust</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shapeFile</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>softImpactSound</td>
<td>VehicleData</td>
</tr>
<tr>
<td>softImpactSpeed</td>
<td>VehicleData</td>
</tr>
<tr>
<td>softSplashSoundVelocity</td>
<td>VehicleData</td>
</tr>
<tr>
<td>splashEmitter</td>
<td>VehicleData</td>
</tr>
<tr>
<td>splashFreqMod</td>
<td>VehicleData</td>
</tr>
<tr>
<td>splashVelEpsilon</td>
<td>VehicleData</td>
</tr>
<tr>
<td>stabDampingConstant</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>stabLenMax</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>stabLenMin</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>stabSpringConstant</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>steeringForce</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>strafeThrustForce</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>triggerDustHeight</td>
<td>VehicleData</td>
</tr>
<tr>
<td>triggerTrailHeight</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>turboFactor</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>underwaterExplosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>useEyePoint</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>vertFactor</td>
<td>HoverVehicleData</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------</td>
</tr>
<tr>
<td>waterWakeSound</td>
<td>VehicleData</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
**HTTPObject Member List**

This is the complete list of members for **HTTPObject**, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>call</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone</td>
<td>SimObject</td>
</tr>
<tr>
<td>connect</td>
<td>TCPObject</td>
</tr>
<tr>
<td>deepClone</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete</td>
<td>SimObject</td>
</tr>
<tr>
<td>disconnect</td>
<td>TCPObject</td>
</tr>
<tr>
<td>dump</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods</td>
<td>SimObject</td>
</tr>
<tr>
<td>get</td>
<td>HTTPObject</td>
</tr>
<tr>
<td>getCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>listen(int port)</td>
<td>TCPObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onConnected()</td>
<td>TCPOObject</td>
</tr>
<tr>
<td>onConnectFailed()</td>
<td>TCPOObject</td>
</tr>
<tr>
<td>onConnectionRequest(string address, string ID)</td>
<td>TCPOObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>onDisconnect()</code></td>
<td>TCPObj ect</td>
</tr>
<tr>
<td><code>onDNSFailed()</code></td>
<td>TCPObj ect</td>
</tr>
<tr>
<td><code>onDNSResolved()</code></td>
<td>TCPObj ect</td>
</tr>
<tr>
<td><code>onLine(string line)</code></td>
<td>TCPObj ect</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObj ect</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObj ect</td>
</tr>
<tr>
<td><code>post(string Address, string requirstURI, string query, string post)</code></td>
<td>HTTPObj ect</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td>SimObj ect</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>SimObj ect</td>
</tr>
<tr>
<td><code>send(string data)</code></td>
<td>TCPObj ect</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td>SimObj ect</td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td>SimObj ect</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObj ect</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>SimObj ect</td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td>SimObj ect</td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td>SimObj ect</td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td>SimObj ect</td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td>SimObj ect</td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td>SimObj ect</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td>SimObj ect</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObj ect</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObj ect</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObj ect</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObj ect</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObj ect</td>
</tr>
</tbody>
</table>
# InteriorInstance Member List

This is the complete list of members for `InteriorInstance`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code> (SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>call</code> (string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>changeMaterial</code> (string mapTo, Material oldMat, Material newMat)</td>
<td>InteriorInstance</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient</code> (NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump</code> (bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>exportToCollada</code> (bool bakeTransform)</td>
<td>InteriorInstance</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getModelFile()</td>
<td>InteriorInstance</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountableObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getNumDetailLevels()</td>
<td>InteriorInstance</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTargetCount(int detailLevel)</code></td>
<td><code>InteriorInstance</code></td>
</tr>
<tr>
<td><code>getTargetName(int detailLevel, int targetNum)</code></td>
<td><code>InteriorInstance</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>interiorFile</code></td>
<td><code>InteriorInstance</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>InteriorInstance [static]</code></td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>InteriorInstance [static]</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setAlarmMode(string alarmMode)</td>
<td>InteriorInstance</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNameSpace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setDetailLevel(int level)</td>
<td>InteriorInstance</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class Name</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>zoneGroup</td>
<td>InteriorInstance</td>
</tr>
</tbody>
</table>
**Item Member List**

This is the complete list of members for *Item*, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>applyDamage(float amount)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>applyImpulse(Point3F pos, Point3F vec)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>GameBase::applyImpulse(Point3F pos, VectorF vel)</td>
<td>GameBase</td>
</tr>
<tr>
<td>applyRadialImpulse(Point3F origin, float radius, float magnitude)</td>
<td>GameBase</td>
</tr>
<tr>
<td>applyRepair(float amount)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>boundingBox</td>
<td>GameBase [static]</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canCloak()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>changeMaterial(string mapTo, Material oldMat, Material newMat)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dataBlock</td>
<td>GameBase</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>destroyThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMeshVisibility()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getAIRepairPoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getControllingClient()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getControllingObject()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageFlash()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamagePercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageState()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDefaultCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEnergyLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEnergyPercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getEyePoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeTransform()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeVector()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getImageAltTrigger(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageAmmo(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageGenericTrigger(int slot, int trigger)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageLoaded(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageScriptAnimPrefix(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageSkinTag(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageState(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageTarget(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageTrigger(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getLastStickyNormal()</code></td>
<td><code>Item</code></td>
</tr>
<tr>
<td><code>getLastStickyPos()</code></td>
<td><code>Item</code></td>
</tr>
<tr>
<td><code>getLookAtPoint(float distance=2000, int typeMask=0xFFFFFFFF)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMaxDamage()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getModelFile()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMountedImage(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountSlot(ShapeBaseImageData)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMuzzlePoint(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMuzzleVector(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getPendingImage(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getRechargeRate()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getRepairRate()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getScale()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getShapeName()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getSkinName()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getSlotTransform(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTargetCount()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getTargetName(int index)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getVelocity()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getWhiteOut()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hasImageState(int slot, string state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isAIControlled</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isAtRest()</td>
<td>Item</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isCloaked()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isDestroyed()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isDisabled()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEnabled()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isHidden()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isImageFiring(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isImageMounted(ShapeBaseImageData image)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
</tbody>
</table>
| isRenderable()                             | Item           | [static]
<table>
<thead>
<tr>
<th>Function/Property</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isRotating()</td>
<td>Item</td>
</tr>
<tr>
<td>isSelectable</td>
<td>Item</td>
</tr>
<tr>
<td>[static]</td>
<td></td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isStatic()</td>
<td>Item</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>[static]</td>
<td></td>
</tr>
<tr>
<td>maxWarpTicks</td>
<td>Item</td>
</tr>
<tr>
<td>minWarpTicks</td>
<td>Item</td>
</tr>
<tr>
<td>mountImage(ShapeBaseImageData image,</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>int slot, bool loaded=true, string skinTag=&quot;&quot;)</td>
<td></td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot,</td>
<td>SceneObject</td>
</tr>
<tr>
<td>TransformF txfm=MatrixF::Identity)</td>
<td></td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onEnterLiquid(string objID, string</td>
<td>Item</td>
</tr>
<tr>
<td>waterCoverage, string liquidType)</td>
<td></td>
</tr>
<tr>
<td>onLeaveLiquid(string objID, string</td>
<td>Item</td>
</tr>
<tr>
<td>liquidType)</td>
<td></td>
</tr>
<tr>
<td>onStickyCollision(string objID)</td>
<td>Item</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pauseThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAudio(int slot, SFXTrack track)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>playThreadId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playThread(int slot, string name=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotate</td>
<td>Item</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setAllMeshesHidden(bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCloaked(bool cloak)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCollisionTimeout(int ignoreColObj=NULL)</td>
<td>Item</td>
</tr>
<tr>
<td>setControl(bool controlled)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setDamageFlash(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageLevel(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageState(string state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageVector(Point3F vec)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDataBlock(GameBaseData data)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEnergyLevel(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>setHidden(bool show)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAltTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAmmo(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageGenericTrigger(int slot, int trigger, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageLoaded(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageScriptAnimPrefix(int slot, string prefix)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTarget(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setInternalName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInvincibleMode(float time, float speed)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setMeshHidden(string name, bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setRechargeRate(float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setRepairRate(float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setShapeName(string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setSkinName(string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setThreadDir(int slot, bool fwd)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setThreadPosition(int slot, float pos)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setThreadTimeScale(int slot, float scale)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setVelocity(Point3F vel)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setWhiteOut(float level)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>skin</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>startFade(int time, int delay, bool fadeOut)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>static</code></td>
<td><code>Item</code></td>
</tr>
<tr>
<td><code>stopAudio(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>stopThread(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountImage(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>validateCameraFov(float fov)</code></td>
<td><code>ShapeBase</code></td>
</tr>
</tbody>
</table>
# ItemData Member List

This is the complete list of members for **ItemData**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>cameraDefaultFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMaxDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMaxFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>category</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>checkDeployPos(TransformF txfm)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>computeCRC</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cubeReflectorDesc</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>Debris</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>debrisShapeName</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>density</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>destroyedLevel</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>disabledLevel</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td>drag</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>elasticity</td>
<td>ItemData</td>
</tr>
<tr>
<td>Explosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>firstPersonOnly</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>friction</td>
<td>ItemData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeployTransform(Point3F pos, Point3F normal)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>gravityMod</td>
<td>ItemData</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>inheritEnergyFromMount</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfClass(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInvincible</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>lightColor</td>
<td>ItemData</td>
</tr>
<tr>
<td>lightOnlyStatic</td>
<td>ItemData</td>
</tr>
<tr>
<td>lightRadius</td>
<td>ItemData</td>
</tr>
<tr>
<td>lightTime</td>
<td>ItemData</td>
</tr>
<tr>
<td>lightType</td>
<td>ItemData</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mass</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>maxDamage</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>maxEnergy</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>maxVelocity</td>
<td>ItemData</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>observeThroughObject</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onAdd(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onCollision(ShapeBase obj, SceneObject)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>Method Name</td>
<td>Arguments</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>collObj, VectorF vec, float len</td>
<td></td>
</tr>
<tr>
<td><strong>onDamage</strong> (ShapeBase obj, float delta)</td>
<td></td>
</tr>
<tr>
<td><strong>onDestroyed</strong> (ShapeBase obj, string lastState)</td>
<td></td>
</tr>
<tr>
<td><strong>onDisabled</strong> (ShapeBase obj, string lastState)</td>
<td></td>
</tr>
<tr>
<td><strong>onEnabled</strong> (ShapeBase obj, string lastState)</td>
<td></td>
</tr>
<tr>
<td><strong>onEndSequence</strong> (ShapeBase obj, int slot)</td>
<td></td>
</tr>
<tr>
<td><strong>onForceUncloak</strong> (ShapeBase obj, string reason)</td>
<td></td>
</tr>
<tr>
<td><strong>onImpact</strong> (ShapeBase obj, SceneObject collObj, VectorF vec, float len)</td>
<td></td>
</tr>
<tr>
<td><strong>onMount</strong> (GameBase obj, SceneObject mountObj, int node)</td>
<td></td>
</tr>
<tr>
<td><strong>onNewDataBlock</strong> (GameBase obj)</td>
<td></td>
</tr>
<tr>
<td><strong>onRemove</strong> (GameBase obj)</td>
<td></td>
</tr>
<tr>
<td><strong>onTrigger</strong> (ShapeBase obj, int index, bool state)</td>
<td></td>
</tr>
<tr>
<td><strong>onUnmount</strong> (GameBase obj, SceneObject mountObj, int node)</td>
<td></td>
</tr>
<tr>
<td><strong>parentGroup</strong></td>
<td></td>
</tr>
<tr>
<td><strong>persistentId</strong></td>
<td></td>
</tr>
<tr>
<td><strong>reloadOnLocalClient</strong></td>
<td></td>
</tr>
<tr>
<td><strong>renderWhenDestroyed</strong></td>
<td></td>
</tr>
<tr>
<td><strong>repairRate</strong></td>
<td></td>
</tr>
<tr>
<td><strong>save</strong> (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td></td>
</tr>
<tr>
<td><strong>schedule</strong> (float time, string method, string args...)</td>
<td></td>
</tr>
<tr>
<td><strong>setCanSave</strong> (bool value=true)</td>
<td></td>
</tr>
<tr>
<td><strong>setClassNamespace</strong> (string name)</td>
<td></td>
</tr>
<tr>
<td><strong>setEditorOnly</strong> (bool value=true)</td>
<td></td>
</tr>
<tr>
<td><strong>setFieldType</strong> (string fieldName, string type)</td>
<td></td>
</tr>
<tr>
<td><strong>setFieldValue</strong> (string fieldName, string value, int</td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>shadowEnable</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowMaxVisibleDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowProjectionDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSize</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSphereAdjust</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shapeFile</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>simpleServerCollision</td>
<td>ItemData</td>
</tr>
<tr>
<td>sticky</td>
<td>ItemData</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>underwaterExplosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>useEyePoint</td>
<td>ShapeBaseData</td>
</tr>
</tbody>
</table>
# LangTable Member List

This is the complete list of members for LangTable, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>addLanguage(string filename,[string languageName])</td>
<td>LangTable</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCurrentLanguage()</td>
<td>LangTable</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getLangName(int language)</td>
<td>LangTable</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getNumLang()</td>
<td>LangTable</td>
</tr>
<tr>
<td>getString(string filename)</td>
<td>LangTable</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>schedule</code> (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCanSave</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCurrentLanguage</code> (int language)</td>
<td>LangTable</td>
</tr>
<tr>
<td><code>setDefaultLanguage</code> (int language)</td>
<td>LangTable</td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>
# LevelInfo Member List

This is the complete list of members for `LevelInfo`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>advancedLightmapSupport</code></td>
<td><code>LevelInfo</code></td>
</tr>
<tr>
<td><code>ambientLightBlendCurve</code></td>
<td><code>LevelInfo</code></td>
</tr>
<tr>
<td><code>ambientLightBlendPhase</code></td>
<td><code>LevelInfo</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canvasClearColor</code></td>
<td><code>LevelInfo</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>decalBias</code></td>
<td><code>LevelInfo</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>fogAtmosphereHeight</code></td>
<td><code>LevelInfo</code></td>
</tr>
<tr>
<td><code>fogColor</code></td>
<td><code>LevelInfo</code></td>
</tr>
<tr>
<td><code>fogDensity</code></td>
<td><code>LevelInfo</code></td>
</tr>
<tr>
<td><code>fogDensityOffset</code></td>
<td><code>LevelInfo</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>nearClip</td>
<td>LevelInfo</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>soundAmbience</code></td>
<td><code>LevelInfo</code></td>
</tr>
<tr>
<td><code>soundDistanceModel</code></td>
<td><code>LevelInfo</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>visibleDistance</code></td>
<td><code>LevelInfo</code></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
**LightAnimData Member List**

This is the complete list of members for `LightAnimData`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code>&lt;br&gt;(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>brightnessA</code></td>
<td>LightAnimData</td>
</tr>
<tr>
<td><code>brightnessKeys</code></td>
<td>LightAnimData</td>
</tr>
<tr>
<td><code>brightnessPeriod</code></td>
<td>LightAnimData</td>
</tr>
<tr>
<td><code>brightnessSmooth</code></td>
<td>LightAnimData</td>
</tr>
<tr>
<td><code>brightnessZ</code></td>
<td>LightAnimData</td>
</tr>
<tr>
<td><code>call</code>(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>colorA</code></td>
<td>LightAnimData</td>
</tr>
<tr>
<td><code>colorKeys</code></td>
<td>LightAnimData</td>
</tr>
<tr>
<td><code>colorPeriod</code></td>
<td>LightAnimData</td>
</tr>
<tr>
<td><code>colorSmooth</code></td>
<td>LightAnimData</td>
</tr>
<tr>
<td><code>colorZ</code></td>
<td>LightAnimData</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump</code>(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------------------------------------------------</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>offsetA</td>
<td>LightAnimData</td>
</tr>
<tr>
<td>offsetKeys</td>
<td>LightAnimData</td>
</tr>
<tr>
<td>offsetPeriod</td>
<td>LightAnimData</td>
</tr>
<tr>
<td>offsetSmooth</td>
<td>LightAnimData</td>
</tr>
<tr>
<td>OffsetZ</td>
<td>LightAnimData</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>rotA</td>
<td>LightAnimData</td>
</tr>
<tr>
<td>rotKeys</td>
<td>LightAnimData</td>
</tr>
<tr>
<td>rotPeriod</td>
<td>LightAnimData</td>
</tr>
<tr>
<td>rotSmooth</td>
<td>LightAnimData</td>
</tr>
<tr>
<td>rotZ</td>
<td>LightAnimData</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td><code>setSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
</tbody>
</table>
**LightBase Member List**

This is the complete list of members for `LightBase`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class/Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>animate</code></td>
<td><code>LightBase</code></td>
</tr>
<tr>
<td><code>animationPeriod</code></td>
<td><code>LightBase</code></td>
</tr>
<tr>
<td><code>animationPhase</code></td>
<td><code>LightBase</code></td>
</tr>
<tr>
<td><code>animationType</code></td>
<td><code>LightBase</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>attenuationRatio</code></td>
<td><code>LightBase</code></td>
</tr>
<tr>
<td><code>brightness</code></td>
<td><code>LightBase</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>castShadows</code></td>
<td><code>LightBase</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>color</code></td>
<td><code>LightBase</code></td>
</tr>
<tr>
<td><code>cookie</code></td>
<td><code>LightBase</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>fadeStartDistance</td>
<td>LightBase</td>
</tr>
<tr>
<td>flareScale</td>
<td>LightBase</td>
</tr>
<tr>
<td>flareType</td>
<td>LightBase</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>includeLightmappedGeometryInShadow</td>
<td>LightBase</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEnabled</td>
<td>LightBase</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>isRenderable</td>
<td>LightBase</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>LightBase</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>lastSplitTerrainOnly</td>
<td>LightBase</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>logWeight</td>
<td>LightBase</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>numSplits</td>
<td>LightBase</td>
</tr>
<tr>
<td>overDarkFactor</td>
<td>LightBase</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pauseAnimation</td>
<td>LightBase</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAnimation(LightAnimData anim)</td>
<td>LightBase</td>
</tr>
<tr>
<td>playAnimation()</td>
<td>LightBase</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>priority</td>
<td>LightBase</td>
</tr>
<tr>
<td>representedInLightmap</td>
<td>LightBase</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool</td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>selectedOnly=false, string preAppendString=&quot;&quot;</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>scale</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>scopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLightEnabled(bool state)</code></td>
<td><code>LightBase</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>shadowDarkenColor</code></td>
<td><code>LightBase</code></td>
</tr>
<tr>
<td><code>shadowDistance</code></td>
<td><code>LightBase</code></td>
</tr>
<tr>
<td><code>shadowSoftness</code></td>
<td><code>LightBase</code></td>
</tr>
<tr>
<td>shadowType</td>
<td>LightBase</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>texSize</td>
<td>LightBase</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# LightDescription Member List

This is the complete list of members for `LightDescription`, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>animationPeriod</td>
<td><code>LightDescription</code></td>
</tr>
<tr>
<td>animationPhase</td>
<td><code>LightDescription</code></td>
</tr>
<tr>
<td>animationType</td>
<td><code>LightDescription</code></td>
</tr>
<tr>
<td>apply()</td>
<td><code>LightDescription</code></td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>attenuationRatio</td>
<td><code>LightDescription</code></td>
</tr>
<tr>
<td>brightness</td>
<td><code>LightDescription</code></td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>canSave</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>castShadows</td>
<td><code>LightDescription</code></td>
</tr>
<tr>
<td>class</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>className</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>clone()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>color</td>
<td><code>LightDescription</code></td>
</tr>
<tr>
<td>cookie</td>
<td><code>LightDescription</code></td>
</tr>
<tr>
<td>deepClone()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>delete()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpMethods()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>fadeStartDistance</td>
<td><code>LightDescription</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>flareScale</td>
<td>LightDescription</td>
</tr>
<tr>
<td>flareType</td>
<td>LightDescription</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>includeLightmappedGeometryInShadow</td>
<td>LightDescription</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>isMethod</code> (string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>lastSplitTerrainOnly</code></td>
<td>LightDescription</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>logWeight</code></td>
<td>LightDescription</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>numSplits</code></td>
<td>LightDescription</td>
</tr>
<tr>
<td><code>overDarkFactor</code></td>
<td>LightDescription</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>range</code></td>
<td>LightDescription</td>
</tr>
<tr>
<td><code>reloadOnLocalClient</code>()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td><code>representedInLightmap</code></td>
<td>LightDescription</td>
</tr>
<tr>
<td><code>save</code> (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule</code> (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCanSave</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>shadowDarkenColor</code></td>
<td>LightDescription</td>
</tr>
<tr>
<td><code>shadowDistance</code></td>
<td>LightDescription</td>
</tr>
<tr>
<td><code>shadowSoftness</code></td>
<td>LightDescription</td>
</tr>
<tr>
<td><code>shadowType</code></td>
<td>LightDescription</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>`texSize</td>
<td>LightDescription</td>
</tr>
</tbody>
</table>
## LightFlareData Member List

This is the complete list of members for **LightFlareData**, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>apply()</td>
<td>LightFlareData</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>elementDist</td>
<td>LightFlareData</td>
</tr>
<tr>
<td>elementRect</td>
<td>LightFlareData</td>
</tr>
<tr>
<td>elementRotate</td>
<td>LightFlareData</td>
</tr>
<tr>
<td>elementScale</td>
<td>LightFlareData</td>
</tr>
<tr>
<td>elementTint</td>
<td>LightFlareData</td>
</tr>
<tr>
<td>elementUseLightColor</td>
<td>LightFlareData</td>
</tr>
<tr>
<td>flareEnabled</td>
<td>LightFlareData</td>
</tr>
<tr>
<td>flareTexture</td>
<td>LightFlareData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>getClassName</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>occlusionRadius</td>
<td>LightFlareData</td>
</tr>
<tr>
<td>overallScale</td>
<td>LightFlareData</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>renderReflectPass</td>
<td>LightFlareData</td>
</tr>
<tr>
<td>save</td>
<td>(string fileNane, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
</tr>
<tr>
<td>schedule</td>
<td>(float time, string method, string args...)</td>
</tr>
<tr>
<td>setCanSave</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setClassNameSpace</td>
<td>(string name)</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setFieldType</td>
<td>(string fieldName, string type)</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>(string fieldName, string value, int index=-1)</td>
</tr>
<tr>
<td>setFilename</td>
<td>(string fileNane)</td>
</tr>
<tr>
<td>setHidden</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setInternalName</td>
<td>(string newInternalName)</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>(bool state=true)</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>(bool state=true)</td>
</tr>
<tr>
<td>setLocked</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setName</td>
<td>(string newName)</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setSuperClassNameSpace</td>
<td>(string name)</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# Lightning Member List

This is the complete list of members for **Lightning**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>applyDamage(Point3F hitPosition, Point3F hitNormal, SceneObject hitObject)</td>
<td>Lightning</td>
</tr>
<tr>
<td>applyImpulse(Point3F pos, VectorF vel)</td>
<td>GameBase</td>
</tr>
<tr>
<td>applyRadialImpulse(Point3F origin, float radius, float magnitude)</td>
<td>GameBase</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>boltStartRadius</td>
<td>Lightning</td>
</tr>
<tr>
<td>boundingBox</td>
<td>GameBase</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>chanceToHitTarget</td>
<td>Lightning</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>color</td>
<td>Lightning</td>
</tr>
<tr>
<td>dataBlock</td>
<td>GameBase</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>fadeColor</code></td>
<td><code>Lightning</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClass_namespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getDataBlock()</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getEulerRotation()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>Lightning</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td>Lightning [static]</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>mountNode</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountPID</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountPos</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountRot</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>position</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>rotation</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>scale</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>scopeToClient(NetConnection client)</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setControl(bool controlled)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>setDataBlock(GameBaseData data)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setScale</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>strikeObject</code></td>
<td><code>Lightning</code></td>
</tr>
<tr>
<td><code>strikeRadius</code></td>
<td><code>Lightning</code></td>
</tr>
<tr>
<td><code>strikeRandomPoint</code></td>
<td><code>Lightning</code></td>
</tr>
<tr>
<td><code>strikesPerMinute</code></td>
<td><code>Lightning</code></td>
</tr>
<tr>
<td><code>strikeWidth</code></td>
<td><code>Lightning</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>unmount</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountObject</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>useFog</code></td>
<td><code>Lightning</code></td>
</tr>
<tr>
<td><code>warningFlashes</code></td>
<td><code>Lightning</code></td>
</tr>
</tbody>
</table>
### LightningData Member List

This is the complete list of members for **LightningData**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>category</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getFieldType</code> (string <code>fieldName</code>)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue</code> (string <code>fieldName</code>, int <code>index=-1</code>)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup</code> (SimGroup <code>group</code>)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField</code> (string <code>fieldName</code>)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code> (string <code>name</code>)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass</code> (string <code>className</code>)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod</code> (string <code>methodName</code>)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>onAdd</code> (GameBase <code>obj</code>)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td><code>onMount</code> (GameBase <code>obj</code>, SceneObject <code>mountObj</code>, int <code>node</code>)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td><code>onNewDataBlock</code> (GameBase <code>obj</code>)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td><code>onRemove</code></td>
<td>GameBaseData</td>
</tr>
<tr>
<td><code>onUnmount</code> (GameBase <code>obj</code>, SceneObject <code>mountObj</code>, int <code>node</code>)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>strikeSound</td>
<td>LightningData</td>
</tr>
<tr>
<td>strikeTextures</td>
<td>LightningData</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>thunderSounds</td>
<td>LightningData</td>
</tr>
</tbody>
</table>
## Marker Member List

This is the complete list of members for Marker, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code> (SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>call</code> (string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient</code> (NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump</code> (bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCanSave</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClientObject</code> ()</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getDebugInfo</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField</code> (int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getEulerRotation</code> ()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td>Marker [static]</td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td>Marker [static]</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>mountNode</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountPID</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountPos</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountRot</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>msToNext</code></td>
<td>Marker</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>position</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>rotation</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>save</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>scale</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>schedule</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>scopeToClient</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>seqNum</code></td>
<td><code>Marker</code></td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setScale</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>smoothingType</code></td>
<td><code>Marker</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>type</code></td>
<td><code>Marker</code></td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountObject</code></td>
<td><code>SceneObject</code></td>
</tr>
</tbody>
</table>
# Material Member List

This is the complete list of members for Material, including all inherited members.

<table>
<thead>
<tr>
<th>Method/Property</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>alphaRef</td>
<td>Material</td>
</tr>
<tr>
<td>alphaTest</td>
<td>Material</td>
</tr>
<tr>
<td>animFlags</td>
<td>Material</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>baseTex</td>
<td>Material</td>
</tr>
<tr>
<td>bumpAtlas</td>
<td>Material</td>
</tr>
<tr>
<td>bumpTex</td>
<td>Material</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>castShadows</td>
<td>Material</td>
</tr>
<tr>
<td>cellIndex</td>
<td>Material</td>
</tr>
<tr>
<td>cellLayout</td>
<td>Material</td>
</tr>
<tr>
<td>cellSize</td>
<td>Material</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>colorMultiply</td>
<td>Material</td>
</tr>
<tr>
<td>cubemap</td>
<td>Material</td>
</tr>
<tr>
<td>customFootstepSound</td>
<td>Material</td>
</tr>
<tr>
<td>customImpactSound</td>
<td>Material</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>detailMap</td>
<td>Material</td>
</tr>
<tr>
<td>detailNormalMap</td>
<td>Material</td>
</tr>
<tr>
<td>detailNormalMapStrength</td>
<td>Material</td>
</tr>
<tr>
<td>detailScale</td>
<td>Material</td>
</tr>
<tr>
<td>detailTex</td>
<td>Material</td>
</tr>
<tr>
<td>diffuseColor</td>
<td>Material</td>
</tr>
<tr>
<td>diffuseMap</td>
<td>Material</td>
</tr>
<tr>
<td>doubleSided</td>
<td>Material</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpInstances</td>
<td>Material</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dynamicCubemap</td>
<td>Material</td>
</tr>
<tr>
<td>effectColor</td>
<td>Material</td>
</tr>
<tr>
<td>emissive</td>
<td>Material</td>
</tr>
<tr>
<td>envMap</td>
<td>Material</td>
</tr>
<tr>
<td>envTex</td>
<td>Material</td>
</tr>
<tr>
<td>flush</td>
<td>Material</td>
</tr>
<tr>
<td>footstepSoundId</td>
<td>Material</td>
</tr>
<tr>
<td>getAnimFlags</td>
<td>Material</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td><code>SimObject::getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getld()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>glow</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>impactSoundId</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isAutoGenerated</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>lightMap</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>mapTo</code></td>
<td><code>Material</code></td>
</tr>
<tr>
<td>minnaertConstant</td>
<td>Material</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>normalMap</td>
<td>Material</td>
</tr>
<tr>
<td>overlayMap</td>
<td>Material</td>
</tr>
<tr>
<td>overlayTex</td>
<td>Material</td>
</tr>
<tr>
<td>parallaxScale</td>
<td>Material</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pixelSpecular</td>
<td>Material</td>
</tr>
<tr>
<td>planarReflection</td>
<td>Material</td>
</tr>
<tr>
<td>reload</td>
<td>Material</td>
</tr>
<tr>
<td>rotPivotOffset</td>
<td>Material</td>
</tr>
<tr>
<td>rotSpeed</td>
<td>Material</td>
</tr>
<tr>
<td><strong>save</strong>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>schedule</strong>(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scrollDir</td>
<td>Material</td>
</tr>
<tr>
<td>scrollSpeed</td>
<td>Material</td>
</tr>
<tr>
<td>sequenceFramePerSec</td>
<td>Material</td>
</tr>
<tr>
<td>sequenceSegmentSize</td>
<td>Material</td>
</tr>
<tr>
<td>setAutoGenerated</td>
<td>Material</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>showDust</td>
<td>Material</td>
</tr>
<tr>
<td>showFootprints</td>
<td>Material</td>
</tr>
<tr>
<td>specular</td>
<td>Material</td>
</tr>
<tr>
<td>specularMap</td>
<td>Material</td>
</tr>
<tr>
<td>specularPower</td>
<td>Material</td>
</tr>
<tr>
<td>subSurface</td>
<td>Material</td>
</tr>
<tr>
<td>subSurfaceColor</td>
<td>Material</td>
</tr>
<tr>
<td>subSurfaceRolloff</td>
<td>Material</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>toneMap</td>
<td>Material</td>
</tr>
<tr>
<td>translucent</td>
<td>Material</td>
</tr>
<tr>
<td>translucentBlendOp</td>
<td>Material</td>
</tr>
<tr>
<td>translucentZWrite</td>
<td>Material</td>
</tr>
<tr>
<td>useAnisotropic</td>
<td>Material</td>
</tr>
<tr>
<td>vertColor</td>
<td>Material</td>
</tr>
<tr>
<td>vertLit</td>
<td>Material</td>
</tr>
<tr>
<td>waveAmp</td>
<td>Material</td>
</tr>
<tr>
<td>waveFreq</td>
<td>Material</td>
</tr>
<tr>
<td>waveType</td>
<td>Material</td>
</tr>
</tbody>
</table>
# MeshRoad Member List

This is the complete list of members for **MeshRoad**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bottomMaterial</td>
<td>MeshRoad</td>
</tr>
<tr>
<td>breakAngle</td>
<td>MeshRoad</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>EditorOpen</td>
<td>MeshRoad</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>MeshRoad [static]</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>MeshRoad [static]</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>Node</td>
<td>MeshRoad</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>postApply()</td>
<td>MeshRoad</td>
</tr>
<tr>
<td>regenerate()</td>
<td>MeshRoad</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(</td>
<td>SimObject</td>
</tr>
<tr>
<td>string fileName, bool</td>
<td></td>
</tr>
<tr>
<td>selectedOnly=false, string</td>
<td></td>
</tr>
<tr>
<td>preAppendString=&quot;&quot; )</td>
<td></td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td></td>
</tr>
<tr>
<td><code>setNodeDepth</code> (int idx, float meters)</td>
<td></td>
</tr>
<tr>
<td><code>setScale</code> (Point3F scale)</td>
<td></td>
</tr>
<tr>
<td><code>setScopeAlways</code> ()</td>
<td></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td></td>
</tr>
<tr>
<td><code>setTransform</code> (TransformF txfm)</td>
<td></td>
</tr>
<tr>
<td><code>showBatches</code></td>
<td></td>
</tr>
<tr>
<td><code>showRoad</code></td>
<td></td>
</tr>
<tr>
<td><code>showSpline</code></td>
<td></td>
</tr>
<tr>
<td><code>sideMaterial</code></td>
<td></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td></td>
</tr>
<tr>
<td><code>textureLength</code></td>
<td></td>
</tr>
<tr>
<td><code>topMaterial</code></td>
<td></td>
</tr>
<tr>
<td><code>unmount</code> ()</td>
<td></td>
</tr>
<tr>
<td><code>unmountObject</code> (SceneObject target)</td>
<td></td>
</tr>
<tr>
<td><code>widthSubdivisions</code></td>
<td></td>
</tr>
<tr>
<td><code>wireframe</code></td>
<td></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# Message Member List

This is the complete list of members for `Message`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>addReference()</code></td>
<td><code>Message</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>freeReference()</code></td>
<td><code>Message</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType()</td>
<td>Message</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAdd()</td>
<td>Message</td>
</tr>
<tr>
<td>onRemove()</td>
<td>Message</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
</tbody>
</table>
# MessageForwarder Member List

This is the complete list of members for `MessageForwarder`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAdd()</td>
<td>ScriptMsgListener</td>
</tr>
<tr>
<td>onAddToQueue(string queue)</td>
<td>ScriptMsgListener</td>
</tr>
<tr>
<td>onMessageObjectReceived(string queue, Message msg)</td>
<td>ScriptMsgListener</td>
</tr>
<tr>
<td>onMessageReceived(string queue, string event, string data)</td>
<td>ScriptMsgListener</td>
</tr>
<tr>
<td>onRemove()</td>
<td>ScriptMsgListener</td>
</tr>
<tr>
<td>onRemoveFromQueue(string queue)</td>
<td>ScriptMsgListener</td>
</tr>
</tbody>
</table>
parentGroup

persistentId

**save**(string fileName, bool selectedOnly=false, string preAppendString="")

**schedule**(float time, string method, string args...)

**setCanSave**(bool value=true)

**setClassNamespace**(string name)

**setEditorOnly**(bool value=true)

**setFieldType**(string fieldName, string type)

**setFieldValue**(string fieldName, string value, int index=-1)

**setFilename**(string fileName)

**setHidden**(bool value=true)

**setInternalName**(string newInternalName)

**setIsExpanded**(bool state=true)

**setIsSelected**(bool state=true)

**setLocked**(bool value=true)

**setName**(string newName)

**setNameChangeAllowed**(bool value=true)

**setSuperClassNamespace**(string name)

**superClass**

**toQueue**

MessageForwarder

Copyright © GarageGames, LLC. All Rights Reserved.
**MessageVector Member List**

This is the complete list of members for `MessageVector`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clear()</code></td>
<td><code>MessageVector</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deleteLine(int deletePos)</code></td>
<td><code>MessageVector</code></td>
</tr>
<tr>
<td><code>dump</code></td>
<td><code>MessageVector</code></td>
</tr>
<tr>
<td><code>dump(string filename, string header)</code></td>
<td><code>MessageVector</code></td>
</tr>
<tr>
<td><code>dump(string filename)</code></td>
<td><code>MessageVector</code></td>
</tr>
<tr>
<td><code>SimObject::dump</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getLineIndexByTag(int tag)</code></td>
<td>MessageVector</td>
</tr>
<tr>
<td><code>getLineTag(int pos)</code></td>
<td>MessageVector</td>
</tr>
<tr>
<td><code>getLineText(int pos)</code></td>
<td>MessageVector</td>
</tr>
<tr>
<td><code>getLineTextByTag(int tag)</code></td>
<td>MessageVector</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getNumLines()</code></td>
<td>MessageVector</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>insertLine(int insertPos, string msg, int tag)</code></td>
<td>MessageVector</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Function/Property</td>
<td>Type</td>
</tr>
<tr>
<td>------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>popBackLine()</td>
<td>MessageVector</td>
</tr>
<tr>
<td>popFrontLine()</td>
<td>MessageVector</td>
</tr>
<tr>
<td>pushBackLine(string msg, int tag)</td>
<td>MessageVector</td>
</tr>
<tr>
<td>pushFrontLine(string msg, int tag)</td>
<td>MessageVector</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>
# MissionArea Member List

This is the complete list of members for MissionArea, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>area</td>
<td>MissionArea</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>flightCeiling</td>
<td>MissionArea</td>
</tr>
<tr>
<td>flightCeilingRange</td>
<td>MissionArea</td>
</tr>
<tr>
<td>getArea()</td>
<td>MissionArea</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>postApply()</td>
<td>MissionArea</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setArea(int x, int y, int width, int height)</td>
<td>MissionArea</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>
# MissionMarker Member List

This is the complete list of members for MissionMarker, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Class Name</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>applyDamage(float amount)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>applyImpulse(Point3F pos, Point3F vec)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>GameBase::applyImpulse(Point3F pos, VectorF vel)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>applyRadialImpulse(Point3F origin, float radius, float magnitude)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>applyRepair(float amount)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>boundingBox</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canCloak()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>changeMaterial(string mapTo, Material oldMat, Material newMat)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dataBlock</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>destroyThread(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>dump</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMeshVisibility</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>dumpMethods</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getAIRepairPoint</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getCameraFov</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getCanSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClientObject</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getControllingClient</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getControllingObject</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getDamageFlash</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getDamageLevel</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getDamagePercent</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getDamageState</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getDataBlock</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>getDebugInfo</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDefaultCameraFov</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getDynamicField</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getEnergyLevel</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getEnergyPercent</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getEulerRotation</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getEyePoint</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getEyeTransform</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getEyeVector()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getImageAltTrigger(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageAmmo(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageGenericTrigger(int slot, int trigger)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageLoaded(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageScriptAnimPrefix(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageSkinTag(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageState(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageTarget(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageTrigger(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getLookAtPoint(float distance=2000, int typeMask=0xFFFFFFFF)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMaxDamage()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getModelFile()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMountedImage(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Method</td>
<td>Object Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountSlot(ShapeBaseImageData)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getMuzzlePoint(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getMuzzleVector(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getPendingImage(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getRechargeRate()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getRepairRate()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setScale()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getShapeName()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getSkinName()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getSlotTransform(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getTargetCount()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getTargetName(int index)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getType()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getVelocity()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td><code>getWhiteOut()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hasImageState(int slot, string state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isAIControlled</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isCloaked()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDestroyed()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDisabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEnabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isHidden()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageFiring(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageMounted(ShapeBaseImageData image)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>MissionMarker</code> [static]</td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td>MissionMarker [static]</td>
</tr>
<tr>
<td><code>isSelected</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isServerObject</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>mountImage</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>mountNode</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountObject</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountPID</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountPos</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountRot</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>pauseThread</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>playAudio</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>playThread</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>position</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>rotation</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>save</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>scopeToClient</code></td>
<td>NetObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>setAllMeshesHidden(bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCloaked(bool cloak)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setControl(bool controlled)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setDamageFlash(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageLevel(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageState(string state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageVector(Point3F vec)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDataBlock(GameBaseData data)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEnergyLevel(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool show)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAltTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAmmo(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageGenericTrigger(int slot, int trigger, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageLoaded(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageScriptAnimPrefix(int slot, string prefix)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTarget(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>setInvincibleMode(float time, float speed)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setMeshHidden(string name, bool hide)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setRechargeRate(float rate)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setRepairRate(float rate)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setName(string name)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setSkinName(string name)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setThreadDir(int slot, bool fwd)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setThreadPosition(int slot, float pos)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setThreadTimeScale(int slot, float scale)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setVelocity(Point3F vel)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setWhiteOut(float level)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>startFade(int time, int delay, bool fadeOut)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>stopAudio(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>stopThread(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>unmountImage(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td>SceneObject</td>
</tr>
</tbody>
</table>
validateCameraFov(float fov)
# MissionMarkerData Member List

This is the complete list of members for `MissionMarkerData`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>cameraDefaultFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMaxDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMaxFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>category</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>checkDeployPos(TransformF txfm)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>computeCRC</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cubeReflectorDesc</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>Debris</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>debrisShapeName</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>density</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>destroyedLevel</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>disabledLevel</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td><code>drag</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>Explosion</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>firstPersonOnly</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeployTransform(Point3F pos, Point3F normal)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>inheritEnergyFromMount</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInvincible</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>locked()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>mass</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>maxDamage</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>maxEnergy</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>observeThroughObject</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>onAdd(GameBase obj)</code></td>
<td>GameBaseData</td>
</tr>
<tr>
<td><code>onCollision(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>onDamage(ShapeBase obj, float delta)</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>onDestroyed(ShapeBase obj, string lastState)</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>onDisabled(ShapeBase obj, string lastState)</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>onEnabled(ShapeBase obj, string lastState)</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>onEndSequence(ShapeBase obj, int slot)</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>onForceUncloak(ShapeBase obj, string reason)</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>onImpact(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>onMount</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onNewDataBlock</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onRemove</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onTrigger</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onUnmount</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>reloadOnLocalClient()</code></td>
<td><code>SimDataBlock</code></td>
</tr>
<tr>
<td><code>renderWhenDestroyed</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>repairRate</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>save</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>setSuperClassNamespace (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>shadowEnable</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowMaxVisibleDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowProjectionDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSize</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSphereAdjust</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shapeFile</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>underwaterExplosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>useEyePoint</td>
<td>ShapeBaseData</td>
</tr>
</tbody>
</table>
NetConnection Member List

This is the complete list of members for **NetConnection**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>checkMaxRate()</td>
<td>NetConnection</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clearPaths()</td>
<td>NetConnection</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>connect(string remoteAddress)</td>
<td>NetConnection</td>
</tr>
<tr>
<td>connectLocal()</td>
<td>NetConnection</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>findObjectByInternalName</code> (string <code>internalName</code>, bool <code>searchChildren=false</code>)</td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getAddress()</code></td>
<td><code>NetConnection</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int </code>index<code>)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getField(int </code>index<code>)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string </code>fieldName<code>)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string </code>fieldName<code>, int </code>index=-1<code>)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getGhostID(int </code>realID<code>)</code></td>
<td><code>NetConnection</code></td>
</tr>
<tr>
<td><code>getGhostsActive()</code></td>
<td><code>NetConnection</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObject(int </code>index<code>)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject </code>obj<code>)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getPacketLoss()</code></td>
<td><code>NetConnection</code></td>
</tr>
<tr>
<td>Method</td>
<td>Module</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>getPing()</td>
<td>NetConnection</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>resolveGhostID(int ghostID)</td>
<td>NetConnection</td>
</tr>
<tr>
<td>resolveObjectFromGhostIndex(int ghostID)</td>
<td>NetConnection</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false,)</td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>preAppendString</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setSimulatedNetParams(float packetLoss, int delay)</code></td>
<td><code>NetConnection</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>sort(string callbackFunction)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>transmitPaths()</code></td>
<td><code>NetConnection</code></td>
</tr>
</tbody>
</table>
# NetObject Member List

This is the complete list of members for **NetObject**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>
## OcclusionVolume Member List

This is the complete list of members for `OcclusionVolume`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code> (SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>call</code> (string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient</code> (NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump</code> (bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>edge</code></td>
<td>OcclusionVolume</td>
</tr>
<tr>
<td><code>getCanSave</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClientObject</code>()</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getDebugInfo</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getEulerRotation()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getScale()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>OcclusionVolume</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>OcclusionVolume</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>plane</td>
<td>OcclusionVolume</td>
</tr>
<tr>
<td>point</td>
<td>OcclusionVolume</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
OpenFileDialog Member List

This is the complete list of members for OpenFileDialog, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>changePath</td>
<td>FileDialog</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>defaultFile</td>
<td>FileDialog</td>
</tr>
<tr>
<td>defaultPath</td>
<td>FileDialog</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Execute()</td>
<td>FileDialog</td>
</tr>
<tr>
<td>fileName</td>
<td>FileDialog</td>
</tr>
<tr>
<td>filters</td>
<td>FileDialog</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(stringfieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getld()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>MultipleFiles</code></td>
<td>OpenFileDialog</td>
</tr>
<tr>
<td><code>MustExist</code></td>
<td>OpenFileDialog</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>title</td>
<td>FileDialog</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# OpenFolderDialog Member List

This is the complete list of members for `OpenFolderDialog`, including all inherited members.

<table>
<thead>
<tr>
<th>Method/Property</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>changePath</code></td>
<td><code>FileDialog</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>defaultFile</code></td>
<td><code>FileDialog</code></td>
</tr>
<tr>
<td><code>defaultPath</code></td>
<td><code>FileDialog</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>Execute()</code></td>
<td><code>FileDialog</code></td>
</tr>
<tr>
<td><code>fileMustExist</code></td>
<td><code>OpenFolderDialog</code></td>
</tr>
<tr>
<td><code>fileName</code></td>
<td><code>FileDialog</code></td>
</tr>
<tr>
<td><code>filters</code></td>
<td><code>FileDialog</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>MultipleFiles</td>
<td>OpenFileDialog</td>
</tr>
<tr>
<td>MustExist</td>
<td>OpenFileDialog</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fileName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>title</td>
<td>FileDialog</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# ParticleData Member List

This is the complete list of members for **ParticleData**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>animateTexture</td>
<td>ParticleData</td>
</tr>
<tr>
<td>animTexFrames</td>
<td>ParticleData</td>
</tr>
<tr>
<td>animTexName</td>
<td>ParticleData</td>
</tr>
<tr>
<td>animTexTiling</td>
<td>ParticleData</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>colors</td>
<td>ParticleData</td>
</tr>
<tr>
<td>constantAcceleration</td>
<td>ParticleData</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dragCoefficient</td>
<td>ParticleData</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>framesPerSec</td>
<td>ParticleData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>gravityCoefficient</td>
<td>ParticleData</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>inheritedVelFactor</td>
<td>ParticleData</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>lifetimeMS</td>
<td>ParticleData</td>
</tr>
<tr>
<td>lifetimeVarianceMS</td>
<td>ParticleData</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>reload()</td>
<td>ParticleData</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>sizes</td>
<td>ParticleData</td>
</tr>
<tr>
<td>spinRandomMax</td>
<td>ParticleData</td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>spinRandomMin</td>
<td>ParticleData</td>
</tr>
<tr>
<td>spinSpeed</td>
<td>ParticleData</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>textureCoords</td>
<td>ParticleData</td>
</tr>
<tr>
<td>textureName</td>
<td>ParticleData</td>
</tr>
<tr>
<td>times</td>
<td>ParticleData</td>
</tr>
<tr>
<td>useInvAlpha</td>
<td>ParticleData</td>
</tr>
<tr>
<td>windCoefficient</td>
<td>ParticleData</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## ParticleEmitter Member List

This is the complete list of members for ParticleEmitter, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>applyImpulse(Point3F pos, VectorF vel)</td>
<td>GameBase</td>
</tr>
<tr>
<td>applyRadialImpulse(Point3F origin, float radius, float magnitude)</td>
<td>GameBase</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>boundingBox</td>
<td>GameBase [static]</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dataBlock</td>
<td>GameBase</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>ParticleEmitter [static]</code></td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>ParticleEmitter [static]</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>mountNode</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td><code>mountObject</code> (SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountPID</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountPos</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountRot</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>position</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>rotation</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>save</code> (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>scale</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>schedule</code> (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>scopeToClient</code> (NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setCanSave</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setControl</code> (bool controlled)</td>
<td>GameBase</td>
</tr>
<tr>
<td><code>setDataBlock</code> (GameBaseData data)</td>
<td>GameBase</td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
</tbody>
</table>
## ParticleEmitterData Member List

This is the complete list of members for `ParticleEmitterData`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>alignDirection</td>
<td><code>ParticleEmitterData</code></td>
</tr>
<tr>
<td>alignParticles</td>
<td><code>ParticleEmitterData</code></td>
</tr>
<tr>
<td>ambientFactor</td>
<td><code>ParticleEmitterData</code></td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>blendStyle</td>
<td><code>ParticleEmitterData</code></td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>canSave</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>category</td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td>class</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>className</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>clone()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>deepClone()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>delete()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpMethods()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>ejectionOffset</td>
<td><code>ParticleEmitterData</code></td>
</tr>
<tr>
<td>ejectionPeriodMS</td>
<td><code>ParticleEmitterData</code></td>
</tr>
<tr>
<td>ejectionVelocity</td>
<td><code>ParticleEmitterData</code></td>
</tr>
<tr>
<td>getCanSave()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>getClassName()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>highResOnly</td>
<td>ParticleEmitterData</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>lifetimeMS</td>
<td>ParticleEmitterData</td>
</tr>
<tr>
<td>Function/Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>lifetimeVarianceMS</td>
<td>ParticleEmitterData</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAdd(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onMount(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onNewDataBlock(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onRemove(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onUnmount(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>orientOnVelocity</td>
<td>ParticleEmitterData</td>
</tr>
<tr>
<td>orientParticles</td>
<td>ParticleEmitterData</td>
</tr>
<tr>
<td>overrideAdvance</td>
<td>ParticleEmitterData</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>particles</td>
<td>ParticleEmitterData</td>
</tr>
<tr>
<td>periodVarianceMS</td>
<td>ParticleEmitterData</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>phiReferenceVel</td>
<td>ParticleEmitterData</td>
</tr>
<tr>
<td>phiVariance</td>
<td>ParticleEmitterData</td>
</tr>
<tr>
<td>reload()</td>
<td>ParticleEmitterData</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>renderReflection</td>
<td>ParticleEmitterData</td>
</tr>
<tr>
<td>reverseOrder</td>
<td>ParticleEmitterData</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>softnessDistance</code></td>
<td><code>ParticleEmitterData</code></td>
</tr>
<tr>
<td><code>sortParticles</code></td>
<td><code>ParticleEmitterData</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>textureName</code></td>
<td><code>ParticleEmitterData</code></td>
</tr>
<tr>
<td><code>thetaMax</code></td>
<td><code>ParticleEmitterData</code></td>
</tr>
<tr>
<td><code>thetaMin</code></td>
<td><code>ParticleEmitterData</code></td>
</tr>
<tr>
<td><code>useEmitterColors</code></td>
<td><code>ParticleEmitterData</code></td>
</tr>
<tr>
<td><code>useEmitterSizes</code></td>
<td><code>ParticleEmitterData</code></td>
</tr>
<tr>
<td><code>velocityVariance</code></td>
<td><code>ParticleEmitterData</code></td>
</tr>
</tbody>
</table>
# ParticleEmitterNode Member List

This is the complete list of members for `ParticleEmitterNode`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Class Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>ParticleEmitterNode</td>
</tr>
<tr>
<td><code>applyImpulse(Point3F pos, VectorF vel)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>applyRadialImpulse(Point3F origin, float radius, float magnitude)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>boundingBox</td>
<td>GameBase</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dataBlock</td>
<td>GameBase</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>emitter</td>
<td>ParticleEmitterNode</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>getClassName</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDataBlock</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>ParticleEmitterNode [static]</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>ParticleEmitterNode [static]</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td><strong>mountObject</strong> (SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td><strong>save</strong> (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td><strong>schedule</strong> (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient (NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setActive (bool active)</td>
<td>ParticleEmitterNode</td>
</tr>
<tr>
<td>setCanSave (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setControl (bool controlled)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setDataBlock (GameBaseData data)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setEditorOnly (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEmitterDataBlock (ParticleEmitterData emitterDatablock=0)</td>
<td>ParticleEmitterNode</td>
</tr>
<tr>
<td>setFieldType (string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setScale</code> (Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setScopeAlways</code> ()</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setTransform</code> (TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>unmount</code> ()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>unmountObject</code> (SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>velocity</code></td>
<td>ParticleEmitterNode</td>
</tr>
</tbody>
</table>
### ParticleEmitterNodeData Member List

This is the complete list of members for `ParticleEmitterNodeData`, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>category</code></td>
<td>GameBaseData</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTitle()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroupName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelect()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAdd(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onMount(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onNewDataBlock(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onRemove(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td><code>onUnmount</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>reloadOnLocalClient()</code></td>
<td><code>SimDataBlock</code></td>
</tr>
<tr>
<td><code>save</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>timeMultiple</code></td>
<td><code>ParticleEmitterNodeData</code></td>
</tr>
</tbody>
</table>
This is the complete list of members for `Path`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>acceptsAsChild(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>add(SimObject objects...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>bringToFront(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>callOnChildren(string method, string args...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>callOnChildrenNoRecurse(string method, string args...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clear()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>findObjectByInternalName</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>Gets the class name.</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>Gets the class namespace.</td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td>Gets the count.</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>Gets the debug info.</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>Gets the declaration line.</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>Gets the dynamic field.</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>Gets the dynamic field count.</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>Gets the field.</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>Gets the field count.</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>Gets the field type.</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>Gets the field value.</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>Gets the filename.</td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td>Gets the full count.</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>Gets the group.</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>Gets the id.</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>Gets the internal name.</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>Gets the name.</td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td>Gets the object.</td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td>Gets the object index.</td>
</tr>
<tr>
<td><code>getPathId()</code></td>
<td>Gets the path id.</td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td>Gets the random.</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>Gets the super class namespace.</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>Gets the hidden.</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>Gets the internal name.</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>Checks if the object is a child of the group.</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>Checks if the object is editor only.</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>Checks if the object is expanded.</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>Checks if the field is a field.</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isLooping</td>
<td>Path</td>
</tr>
<tr>
<td>isMember(obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onObjectAdded(object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pushToBack(obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(child1, child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>save(fileName, selectedOnly=false, preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(time, method, args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(fieldName, type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(fieldValue, fieldValue, index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>sort</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
</tbody>
</table>
PathCamera Member List

This is the complete list of members for PathCamera, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>applyDamage</strong> (float amount)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><strong>applyImpulse</strong> (Point3F pos, Point3F vec)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><strong>GameBase::applyImpulse</strong> (Point3F pos, VectorF vel)</td>
<td>GameBase</td>
</tr>
<tr>
<td><strong>applyRadialImpulse</strong> (Point3F origin, float radius, float magnitude)</td>
<td>GameBase</td>
</tr>
<tr>
<td><strong>applyRepair</strong> (float amount)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><strong>assignFieldsFrom</strong> (SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>assignPersistentId</strong> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>boundingBox</strong></td>
<td>GameBase [static]</td>
</tr>
<tr>
<td><strong>call</strong> (string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>canCloak</strong></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><strong>canSave</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>canSaveDynamicFields</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>changeMaterial</strong> (string mapTo, Material oldMat, Material newMat)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><strong>class</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>className</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>clearScopeToClient</strong> (NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td><strong>clone</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>dataBlock</strong></td>
<td>GameBase</td>
</tr>
<tr>
<td><strong>deepClone</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>delete</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>destroyThread</strong> (int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><strong>dump</strong> (bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMeshVisibility()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getAIRepairPoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getControllingClient()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getControllingObject()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageFlash()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamagePercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageState()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDefaultCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEnergyLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEnergyPercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getEyePoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeTransform()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeVector()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getImageAltTrigger(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageAmmo(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageGenericTrigger(int slot, int trigger)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageLoaded(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageScriptAnimPrefix(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageSkinTag(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageState(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageTarget(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageTrigger(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getLookAtPoint(float distance=2000, int typeMask=0xFFFFFFFF)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMaxDamage()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getModelFile()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMountedImage(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountSlot(ShapeBaseImageData)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getMuzzlePoint(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getMuzzleVector(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getPendingImage(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getRechargeRate()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getRepairRate()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getScale()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getShapeName()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getSkinName()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getSlotTransform(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getTargetCount()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getTargetName(int index)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getType()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getVelocity()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getWhiteOut()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hasImageState(int slot, string state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isAIControlled</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isCloaked()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDestroyed()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDisabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEnabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isHidden()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageFiring(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageMounted(ShapeBaseImageData image)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>PathCamera [static]</code></td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>PathCamera [static]</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountImage(ShapeBaseImageData image, int slot, bool loaded=true, string skinTag=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onNode(string node)</td>
<td>PathCamera</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pauseThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAudio(int slot, SFXTrack track)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>playThread(int slot, string name=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>popFront()</td>
<td>PathCamera</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>pushBack(TransformF transform, float speed=1.0, string type=&quot;Normal&quot;, string path=&quot;Linear&quot;)</td>
<td>PathCamera</td>
</tr>
<tr>
<td>pushFront(TransformF transform, float speed=1.0, string type=&quot;Normal&quot;, string path=&quot;Linear&quot;)</td>
<td>PathCamera</td>
</tr>
<tr>
<td>reset(float speed=1.0f)</td>
<td>PathCamera</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Object Type</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><code>scale</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>scopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setAllMeshesHidden(bool hide)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setCameraFov(float fov)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCloaked(bool cloak)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setControl(bool controlled)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>setDamageFlash(float level)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setDamageLevel(float level)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setDamageState(string state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setDamageVector(Point3F vec)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setDataBlock(GameBaseData data)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEnergyLevel(float level)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden(bool show)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageAltTrigger(int slot, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageAmmo(int slot, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageGenericTrigger(int slot, int trigger, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageLoaded(int slot, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageScriptAnimPrefix(int slot, string</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>setImageTarget</code> (int slot, bool state)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageTrigger</code> (int slot, bool state)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInvincibleMode</code> (float time, float speed)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setMeshHidden</code> (string name, bool hide)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setPosition</code> (float position=0.0f)</td>
<td><code>PathCamera</code></td>
</tr>
<tr>
<td><code>setRechargeRate</code> (float rate)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setRepairRate</code> (float rate)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setScale</code> (Point3F scale)</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways</code> ()</td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setShapeName</code> (string name)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setSkinName</code> (string name)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setState</code> (string newState=&quot;forward&quot;)</td>
<td><code>PathCamera</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTarget</code> (float position=1.0f)</td>
<td><code>PathCamera</code></td>
</tr>
<tr>
<td><code>setThreadDir</code> (int slot, bool fwd)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setThreadPosition</code> (int slot, float pos)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setThreadTimeScale</code> (int slot, float scale)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setTransform</code> (TransformF txfm)</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setVelocity</code> (Point3F vel)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setWhiteOut</code> (float level)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>skin</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>startFade</code> (int time, int delay, bool fadeOut)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>stopAudio</code> (int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>stopThread</code> (int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>unmount</code> ()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>unmountImage</code> (int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>unmountObject</code> (SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>validateCameraFov</code> (float fov)</td>
<td>ShapeBase</td>
</tr>
</tbody>
</table>
**PathCameraData Member List**

This is the complete list of members for *PathCameraData*, including all inherited members.

<table>
<thead>
<tr>
<th>Method/Property</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>cameraDefaultFov</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cameraMaxDist</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cameraMaxFov</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cameraMinDist</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cameraMinFov</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>category</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>checkDeployPos(TransformF txfm)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>computeCRC</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cubeReflectorDesc</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>Debris</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>debrisShapeName</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>density</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>destroyedLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>disabledLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>drag</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Explosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>firstPersonOnly</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeployTransform(Point3F pos, Point3F normal)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>inheritEnergyFromMount</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class Name</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInvincible</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mass</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>maxDamage</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>maxEnergy</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>observeThroughObject</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onAdd(GameBase obj)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onCollision(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onDamage(ShapeBase obj, float delta)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onDestroyed(ShapeBase obj, string lastState)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onDisabled(ShapeBase obj, string lastState)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onEnabled(ShapeBase obj, string lastState)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onEndSequence(ShapeBase obj, int slot)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onForceUncloak(ShapeBase obj, string reason)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onImpact(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>onMount</td>
<td>(GameBase obj, SceneObject mountObj, int node)</td>
</tr>
<tr>
<td>onNewDataBlock</td>
<td>(GameBase obj)</td>
</tr>
<tr>
<td>onRemove</td>
<td>(GameBase obj)</td>
</tr>
<tr>
<td>onTrigger</td>
<td>(ShapeBase obj, int index, bool state)</td>
</tr>
<tr>
<td>onUnmount</td>
<td>(GameBase obj, SceneObject mountObj, int node)</td>
</tr>
<tr>
<td>parentGroup</td>
<td></td>
</tr>
<tr>
<td>persistentId</td>
<td></td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td></td>
</tr>
<tr>
<td>renderWhenDestroyed</td>
<td></td>
</tr>
<tr>
<td>repairRate</td>
<td></td>
</tr>
<tr>
<td>save</td>
<td>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
</tr>
<tr>
<td>schedule</td>
<td>(float time, string method, string args...)</td>
</tr>
<tr>
<td>setCanSave</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setClassNameSpace</td>
<td>(string name)</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setFieldType</td>
<td>(string fieldName, string type)</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>(string fieldName, string value, int index=-1)</td>
</tr>
<tr>
<td>setFilename</td>
<td>(string fileName)</td>
</tr>
<tr>
<td>setHidden</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setInternalName</td>
<td>(string newInternalName)</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>(bool state=true)</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>(bool state=true)</td>
</tr>
<tr>
<td>setLocked</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setName</td>
<td>(string newName)</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setSuperClassNamespace (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>shadowEnable</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowMaxVisibleDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowProjectionDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSize</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSphereAdjust</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shapeFile</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>underwaterExplosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>useEyePoint</td>
<td>ShapeBaseData</td>
</tr>
</tbody>
</table>
# PfxVis Member List

This is the complete list of members for **PfxVis**, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>clear()</code></td>
<td>PfxVis</td>
</tr>
<tr>
<td><code>hide()</code></td>
<td>PfxVis</td>
</tr>
<tr>
<td><code>onWindowClosed(GuiWindowCtrl *ctrl)</code></td>
<td>PfxVis</td>
</tr>
<tr>
<td><code>open(PostEffect effect, bool clear)</code></td>
<td>PfxVis</td>
</tr>
<tr>
<td><code>show()</code></td>
<td>PfxVis</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
### PhysicalZone Member List

This is the complete list of members for `PhysicalZone`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>activate()</td>
<td>PhysicalZone</td>
</tr>
<tr>
<td>appliedForce</td>
<td>PhysicalZone</td>
</tr>
<tr>
<td>assignFieldsFrom( SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call( string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient( NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deactivate()</td>
<td>PhysicalZone</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump( bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>gravityMod</td>
<td>PhysicalZone</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>PhysicalZone</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>PhysicalZone</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>polyhedron</td>
<td>PhysicalZone</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>renderZones</td>
<td>PhysicalZone</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked</td>
<td>SimObject</td>
</tr>
</tbody>
</table>

[static]
<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>velocityMod</code></td>
<td><code>PhysicalZone</code></td>
</tr>
</tbody>
</table>
# PhysicsDebris Member List

This is the complete list of members for `PhysicsDebris`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>applyImpulse(Point3F pos, VectorF vel)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>applyRadialImpulse(Point3F origin, float radius, float magnitude)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>boundingBox</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dataBlock</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>PhysicsDebris  [static]</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>PhysicsDebris  [static]</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setControl(bool controlled)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setDataBlock(GameBaseData data)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td>SceneObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# PhysicsDebrisData Member List

This is the complete list of members for `PhysicsDebrisData`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>angularDamping</td>
<td><code>PhysicsDebrisData</code></td>
</tr>
<tr>
<td>angularSleepThreshold</td>
<td><code>PhysicsDebrisData</code></td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>buoyancyDensity</td>
<td><code>PhysicsDebrisData</code></td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>canSave</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>castShadows</td>
<td><code>PhysicsDebrisData</code></td>
</tr>
<tr>
<td>category</td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td>class</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>className</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>clone()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>deepClone()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>delete()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpMethods()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>friction</td>
<td><code>PhysicsDebrisData</code></td>
</tr>
<tr>
<td>getCanSave()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>getClassName()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>lifetime</td>
<td>PhysicsDebrisData</td>
</tr>
<tr>
<td>lifetimeVariance</td>
<td>PhysicsDebrisData</td>
</tr>
<tr>
<td>linearDamping</td>
<td>PhysicsDebrisData</td>
</tr>
<tr>
<td>linearSleepThreshold</td>
<td>PhysicsDebrisData</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mass</td>
<td>PhysicsDebrisData</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>onAdd</strong>(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td><strong>onMount</strong>(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td><strong>onNewDataBlock</strong>(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td><strong>onRemove</strong>(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td><strong>onUnmount</strong>(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>preload</td>
<td>PhysicsDebrisData</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>restitution</td>
<td>PhysicsDebrisData</td>
</tr>
<tr>
<td><strong>save</strong>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>schedule</strong>(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setCanSave</strong>(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setClassNamespace</strong>(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setEditorOnly</strong>(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setFieldType</strong>(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setFieldValue</strong>(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setFilename</strong>(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setHidden</strong>(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setInternalName</strong>(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>shapeFile</code></td>
<td><code>PhysicsDebrisData</code></td>
</tr>
<tr>
<td><code>staticFriction</code></td>
<td><code>PhysicsDebrisData</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>waterDampingScale</code></td>
<td><code>PhysicsDebrisData</code></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# PhysicsForce Member List

This is the complete list of members for **PhysicsForce**, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code> (SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>attach</code> (Point3F start, Point3F direction, float maxDist)</td>
<td>PhysicsForce</td>
</tr>
<tr>
<td><code>call</code> (string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient</code> (NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>detach</code> (Point3F force=Point3F::Zero)</td>
<td>PhysicsForce</td>
</tr>
<tr>
<td><code>dump</code> (bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCanSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClientObject</code> ()</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getDebugInfo</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Context</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isAttached()</td>
<td>PhysicsForce</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderer[static]</td>
<td>PhysicsForce</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable[static]</td>
<td>PhysicsForce</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>----------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td><code>SceneObject</code></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# PhysicsShape Member List

This is the complete list of members for `PhysicsShape`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>applyImpulse(Point3F pos, VectorF vel)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>applyRadialImpulse(Point3F origin, float radius, float magnitude)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>boundingBox</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dataBlock</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>destroy()</code></td>
<td><code>PhysicsShape</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>addClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>getScale()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getType()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>isDestroyed()</code></td>
<td>PhysicsShape</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td>PhysicsShape</td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td>PhysicsShape</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>noCorrections</td>
<td>PhysicsShape</td>
</tr>
<tr>
<td>noSmoothing</td>
<td>PhysicsShape</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAmbient</td>
<td>PhysicsShape</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>restore()</td>
<td>PhysicsShape</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setControl(bool controlled)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setDataBlock(GameBaseData data)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>SimObject</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td></td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setScale</code> (Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setScopeAlways</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setTransform</code> (TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>unmount</code> ()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>unmountObject</code> (SceneObject target)</td>
<td>SceneObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# PhysicsShapeData Member List

This is the complete list of members for `PhysicsShapeData`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>angularDamping</td>
<td>PhysicsShapeData</td>
</tr>
<tr>
<td>angularSleepThreshold</td>
<td>PhysicsShapeData</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>buoyancyDensity</td>
<td>PhysicsShapeData</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>category</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Debris</td>
<td>PhysicsShapeData</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>destroyedShape</td>
<td>PhysicsShapeData</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Explosion</td>
<td>PhysicsShapeData</td>
</tr>
<tr>
<td>friction</td>
<td>PhysicsShapeData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>linearDamping</td>
<td>PhysicsShapeData</td>
</tr>
<tr>
<td>linearSleepThreshold</td>
<td>PhysicsShapeData</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mass</td>
<td>PhysicsShapeData</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAdd(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onMount(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onNewDataBlock(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onRemove(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onUnmount(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>restitution</td>
<td>PhysicsShapeData</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>shapeName</code></td>
<td><code>PhysicsShapeData</code></td>
</tr>
<tr>
<td><code>simType</code></td>
<td><code>PhysicsShapeData</code></td>
</tr>
<tr>
<td><code>staticFriction</code></td>
<td><code>PhysicsShapeData</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>waterDampingScale</code></td>
<td><code>PhysicsShapeData</code></td>
</tr>
</tbody>
</table>
## Player Member List

This is the complete list of members for `Player`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>allowAllPoses()</code></td>
<td><code>Player</code></td>
</tr>
<tr>
<td><code>allowCrouching(bool state)</code></td>
<td><code>Player</code></td>
</tr>
<tr>
<td><code>allowJetJumping(bool state)</code></td>
<td><code>Player</code></td>
</tr>
<tr>
<td><code>allowJumping(bool state)</code></td>
<td><code>Player</code></td>
</tr>
<tr>
<td><code>allowProne(bool state)</code></td>
<td><code>Player</code></td>
</tr>
<tr>
<td><code>allowSprinting(bool state)</code></td>
<td><code>Player</code></td>
</tr>
<tr>
<td><code>allowSwimming(bool state)</code></td>
<td><code>Player</code></td>
</tr>
<tr>
<td><code>applyDamage(float amount)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>applyImpulse(Point3F pos, Point3F vec)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>GameBase::applyImpulse(Point3F pos, VectorF vel)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>applyRadialImpulse(Point3F origin, float radius, float magnitude)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>applyRepair(float amount)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>boundingBox</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canCloak()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>changeMaterial(string mapTo, Material oldMat, Material newMat)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>checkDismountPoint(Point3F oldPos, Point3F pos)</code></td>
<td><code>Player</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clearControlObject()</code></td>
<td><code>Player</code></td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>crouchTrigger</code></td>
<td><code>Player</code></td>
</tr>
<tr>
<td><code>dataBlock</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>destroyThread(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMeshVisibility()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getAIRepairPoint()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getCameraFov()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNameSpace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getControllingClient()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getControllingObject()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getControlObject()</code></td>
<td><code>Player</code></td>
</tr>
<tr>
<td><code>getDamageFlash()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamageLevel()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamageLocation(Point3F pos)</code></td>
<td><code>Player</code></td>
</tr>
<tr>
<td><code>getDamagePercent()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>getDamageState()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDefaultCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEnergyLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEnergyPercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getEyePoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeTransform()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeVector()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getImageAltTrigger(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageAmmo(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageGenericTrigger(int slot, int trigger)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageLoaded(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageScriptAnimPrefix(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>getImageSkinTag(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageState(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageTarget(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageTrigger(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getLookAtPoint(float distance=2000, int typeMask=0xFFFFFFFF)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMaxDamage()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getModelFile()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountSlot(ShapeBaseImageData image)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzlePoint(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzleVector(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getNumDeathAnimations()</td>
<td>Player</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPendingImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getPose()</td>
<td>Player</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRechargeRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRepairRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getScale()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getShapeName()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getSkinName()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getSlotTransform(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getState()</code></td>
<td><code>Player</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTargetCount()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getTargetName(int index)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getVelocity()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getWhiteOut()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hasImageState(int slot, string state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>imageTrigger0</code></td>
<td><code>Player</code></td>
</tr>
<tr>
<td><code>imageTrigger1</code></td>
<td><code>Player</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isAIControlled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isCloaked()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDestroyed()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDisabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEnabled()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isHidden()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isImageFiring(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isImageMounted(ShapeBaseImageData image)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>Player [static]</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>Player [static]</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>jumpJetTrigger</td>
<td>Player [static]</td>
</tr>
<tr>
<td>jumpTrigger</td>
<td>Player [static]</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>maxImpulseVelocity</td>
<td>Player [static]</td>
</tr>
<tr>
<td>maxPredictionTicks</td>
<td>Player [static]</td>
</tr>
<tr>
<td>maxWarpTicks</td>
<td>Player [static]</td>
</tr>
<tr>
<td>minWarpTicks</td>
<td>Player [static]</td>
</tr>
<tr>
<td>mountImage(ShapeBaseImageData image,)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Function/Variable</td>
<td>Type</td>
</tr>
<tr>
<td>--------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>int slot, bool loaded=true, string skinTag=&quot;&quot;</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pauseThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAudio(int slot, SFXTrack track)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>playThread(int slot, string name=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>proneTrigger</td>
<td>Player</td>
</tr>
<tr>
<td>renderCollision</td>
<td>Player</td>
</tr>
<tr>
<td>renderMyItems</td>
<td>Player</td>
</tr>
<tr>
<td>renderMyPlayer</td>
<td>Player</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setActionThread(string name, bool hold=false, bool fsp=true)</td>
<td>Player</td>
</tr>
<tr>
<td>setAllMeshesHidden(bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>setArmThread</td>
<td>(string name)</td>
</tr>
<tr>
<td>setCameraFov</td>
<td>(float fov)</td>
</tr>
<tr>
<td>setCanSave</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>(string name)</td>
</tr>
<tr>
<td>setCloaked</td>
<td>(bool cloak)</td>
</tr>
<tr>
<td>setControl</td>
<td>(bool controlled)</td>
</tr>
<tr>
<td>setControlObject</td>
<td>(ShapeBase obj)</td>
</tr>
<tr>
<td>setDamageFlash</td>
<td>(float level)</td>
</tr>
<tr>
<td>setDamageLevel</td>
<td>(float level)</td>
</tr>
<tr>
<td>setDamageState</td>
<td>(string state)</td>
</tr>
<tr>
<td>setDamageVector</td>
<td>(Point3F vec)</td>
</tr>
<tr>
<td>setDataBlock</td>
<td>(GameBaseData data)</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setEnergyLevel</td>
<td>(float level)</td>
</tr>
<tr>
<td>setFieldType</td>
<td>(string fieldName, string type)</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>(string fieldName, string value, int index=-1)</td>
</tr>
<tr>
<td>setFilename</td>
<td>(string fileName)</td>
</tr>
<tr>
<td>setHidden</td>
<td>(bool show)</td>
</tr>
<tr>
<td>setImageAltTrigger</td>
<td>(int slot, bool state)</td>
</tr>
<tr>
<td>setImageAmmo</td>
<td>(int slot, bool state)</td>
</tr>
<tr>
<td>setImageGenericTrigger</td>
<td>(int slot, int trigger, bool state)</td>
</tr>
<tr>
<td>setImageLoaded</td>
<td>(int slot, bool state)</td>
</tr>
<tr>
<td>setImageScriptAnimPrefix</td>
<td>(int slot, string prefix)</td>
</tr>
<tr>
<td>setImageTarget</td>
<td>(int slot, bool state)</td>
</tr>
<tr>
<td>setImageTrigger</td>
<td>(int slot, bool state)</td>
</tr>
</tbody>
</table>

**Classes:**
- Player
- ShapeBase
- SimObject
- GameBase
<table>
<thead>
<tr>
<th>Function</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>setInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInvincibileMode</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setMeshHidden</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setName</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setRechargeRate</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setRepairRate</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setScale</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setShapeName</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setSkinName</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setThreadDir</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setThreadPosition</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setThreadTimeScale</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setTransform</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setVelocity</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setWhiteOut</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>skin</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>sprintTrigger</code></td>
<td><code>Player</code></td>
</tr>
<tr>
<td><code>startFade</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>stopAudio</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>stopThread</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>validateCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>vehicleDismountTrigger</td>
<td>Player</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
**PlayerData Member List**

This is the complete list of members for `PlayerData`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>airControl</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>allowImageStateAnimation</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>animationDone(Player obj)</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>boundingBox</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>boxHeadBackPercentage</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>boxHeadFrontPercentage</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>boxHeadLeftPercentage</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>boxHeadPercentage</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>boxHeadRightPercentage</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>boxTorsoPercentage</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>bubbleEmitTime</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>cameraDefaultFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMaxDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMaxFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>category</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>checkDeployPos(TransformF tfm)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Variable/Function</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>computeCRC</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>crouchBoundingBox</td>
<td>PlayerData</td>
</tr>
<tr>
<td>crouchForce</td>
<td>PlayerData</td>
</tr>
<tr>
<td>cubeReflectorDesc</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>Debris</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>debrisShapeName</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>DecalData</td>
<td>PlayerData</td>
</tr>
<tr>
<td>decalOffset</td>
<td>PlayerData</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>density</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>destroyedLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>disabledLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>doDismount(Player obj)</td>
<td>PlayerData</td>
</tr>
<tr>
<td>drag</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dustEmitter</td>
<td>PlayerData</td>
</tr>
<tr>
<td>exitingWater</td>
<td>PlayerData</td>
</tr>
<tr>
<td>exitSplashSoundVelocity</td>
<td>PlayerData</td>
</tr>
<tr>
<td>Explosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>fallingSpeedThreshold</td>
<td>PlayerData</td>
</tr>
<tr>
<td>firstPersonOnly</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>firstPersonShadows</td>
<td>PlayerData</td>
</tr>
<tr>
<td>FootBubblesSound</td>
<td>PlayerData</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>FootHardSound</td>
<td>PlayerData</td>
</tr>
<tr>
<td>FootMetalSound</td>
<td>PlayerData</td>
</tr>
<tr>
<td>footPuffEmitter</td>
<td>PlayerData</td>
</tr>
<tr>
<td>footPuffNumParts</td>
<td>PlayerData</td>
</tr>
<tr>
<td>footPuffRadius</td>
<td>PlayerData</td>
</tr>
<tr>
<td>FootShallowSound</td>
<td>PlayerData</td>
</tr>
<tr>
<td>FootSnowSound</td>
<td>PlayerData</td>
</tr>
<tr>
<td>FootSoftSound</td>
<td>PlayerData</td>
</tr>
<tr>
<td>footstepSplashHeight</td>
<td>PlayerData</td>
</tr>
<tr>
<td>FootUnderwaterSound</td>
<td>PlayerData</td>
</tr>
<tr>
<td>FootWadingSound</td>
<td>PlayerData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeployTransform()</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>groundImpactMinSpeed</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>groundImpactShakeAmp</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>groundImpactShakeDuration</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>groundImpactShakeFalloff</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>groundImpactShakeFreq</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>hardSplashSoundVelocity</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>horizMaxSpeed</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>horizResistFactor</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>horizResistSpeed</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>imageAnimPrefix</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>imageAnimPrefixFP</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>impactHardSound</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>impactMetalSound</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>impactSnowSound</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>impactSoftSound</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>impactWaterEasy</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>impactWaterHard</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>impactWaterMedium</code></td>
<td><code>PlayerData</code></td>
</tr>
<tr>
<td><code>inheritEnergyFromMount</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function/Property</td>
<td>Class/Type</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInvincible</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>isMemberOfClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>jetJumpEnergyDrain</code></td>
<td>PlayerData</td>
</tr>
<tr>
<td><code>jetJumpForce</code></td>
<td>PlayerData</td>
</tr>
<tr>
<td><code>jetJumpSurfaceAngle</code></td>
<td>PlayerData</td>
</tr>
<tr>
<td><code>jetMaxJumpSpeed</code></td>
<td>PlayerData</td>
</tr>
<tr>
<td><code>jetMinJumpEnergy</code></td>
<td>PlayerData</td>
</tr>
<tr>
<td><code>jetMinJumpSpeed</code></td>
<td>PlayerData</td>
</tr>
<tr>
<td><code>jumpDelay</code></td>
<td>PlayerData</td>
</tr>
<tr>
<td><code>jumpEnergyDrain</code></td>
<td>PlayerData</td>
</tr>
<tr>
<td><code>jumpForce</code></td>
<td>PlayerData</td>
</tr>
<tr>
<td><code>jumpSurfaceAngle</code></td>
<td>PlayerData</td>
</tr>
<tr>
<td><code>jumpTowardsNormal</code></td>
<td>PlayerData</td>
</tr>
<tr>
<td><code>landSequenceTime</code></td>
<td>PlayerData</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>mass</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>maxBackwardSpeed</code></td>
<td>PlayerData</td>
</tr>
<tr>
<td><code>maxCrouchBackwardSpeed</code></td>
<td>PlayerData</td>
</tr>
<tr>
<td><code>maxCrouchForwardSpeed</code></td>
<td>PlayerData</td>
</tr>
<tr>
<td><code>maxCrouchSideSpeed</code></td>
<td>PlayerData</td>
</tr>
<tr>
<td><code>maxDamage</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>maxEnergy</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>maxForwardSpeed</code></td>
<td>PlayerData</td>
</tr>
<tr>
<td><code>maxFreelookAngle</code></td>
<td>PlayerData</td>
</tr>
<tr>
<td>Variable</td>
<td>Data Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>maxJumpSpeed</td>
<td>PlayerData</td>
</tr>
<tr>
<td>maxLookAngle</td>
<td>PlayerData</td>
</tr>
<tr>
<td>maxProneBackwardSpeed</td>
<td>PlayerData</td>
</tr>
<tr>
<td>maxProneForwardSpeed</td>
<td>PlayerData</td>
</tr>
<tr>
<td>maxProneSideSpeed</td>
<td>PlayerData</td>
</tr>
<tr>
<td>maxSideSpeed</td>
<td>PlayerData</td>
</tr>
<tr>
<td>maxSprintBackwardSpeed</td>
<td>PlayerData</td>
</tr>
<tr>
<td>maxSprintForwardSpeed</td>
<td>PlayerData</td>
</tr>
<tr>
<td>maxSprintSideSpeed</td>
<td>PlayerData</td>
</tr>
<tr>
<td>maxStepHeight</td>
<td>PlayerData</td>
</tr>
<tr>
<td>maxTimeScale</td>
<td>PlayerData</td>
</tr>
<tr>
<td>maxUnderwaterBackwardSpeed</td>
<td>PlayerData</td>
</tr>
<tr>
<td>maxUnderwaterForwardSpeed</td>
<td>PlayerData</td>
</tr>
<tr>
<td>maxUnderwaterSideSpeed</td>
<td>PlayerData</td>
</tr>
<tr>
<td>mediumSplashSoundVelocity</td>
<td>PlayerData</td>
</tr>
<tr>
<td>minImpactSpeed</td>
<td>PlayerData</td>
</tr>
<tr>
<td>minJumpEnergy</td>
<td>PlayerData</td>
</tr>
<tr>
<td>minJumpSpeed</td>
<td>PlayerData</td>
</tr>
<tr>
<td>minLaterallImpactSpeed</td>
<td>PlayerData</td>
</tr>
<tr>
<td>minLookAngle</td>
<td>PlayerData</td>
</tr>
<tr>
<td>minRunEnergy</td>
<td>PlayerData</td>
</tr>
<tr>
<td>minSprintEnergy</td>
<td>PlayerData</td>
</tr>
<tr>
<td>movingBubblesSound</td>
<td>PlayerData</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>observeThroughObject</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onAdd(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onCollision(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>Event Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>onDamage</td>
<td>(ShapeBase obj, float delta)</td>
</tr>
<tr>
<td>onDestroyed</td>
<td>(ShapeBase obj, string lastState)</td>
</tr>
<tr>
<td>onDisabled</td>
<td>(ShapeBase obj, string lastState)</td>
</tr>
<tr>
<td>onEnabled</td>
<td>(ShapeBase obj, string lastState)</td>
</tr>
<tr>
<td>onEndSequence</td>
<td>(ShapeBase obj, int slot)</td>
</tr>
<tr>
<td>onEnterLiquid</td>
<td>(Player obj, float coverage, string type)</td>
</tr>
<tr>
<td>onEnterMissionArea</td>
<td>(Player obj)</td>
</tr>
<tr>
<td>onForceUncloak</td>
<td>(ShapeBase obj, string reason)</td>
</tr>
<tr>
<td>onImpact</td>
<td>(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</td>
</tr>
<tr>
<td>onLeaveLiquid</td>
<td>(Player obj, string type)</td>
</tr>
<tr>
<td>onLeaveMissionArea</td>
<td>(Player obj)</td>
</tr>
<tr>
<td>onMount</td>
<td>(GameBase obj, SceneObject mountObj, int node)</td>
</tr>
<tr>
<td>onNewDataBlock</td>
<td>(GameBase obj)</td>
</tr>
<tr>
<td>onPoseChange</td>
<td>(Player obj, string oldPose, string newPose)</td>
</tr>
<tr>
<td>onRemove</td>
<td>(GameBase obj)</td>
</tr>
<tr>
<td>onStartSprintMotion</td>
<td>(Player obj)</td>
</tr>
<tr>
<td>onStartSwim</td>
<td>(Player obj)</td>
</tr>
<tr>
<td>onStopSprintMotion</td>
<td>(Player obj)</td>
</tr>
<tr>
<td>onStopSwim</td>
<td>(Player obj)</td>
</tr>
<tr>
<td>onTrigger</td>
<td>(ShapeBase obj, int index, bool state)</td>
</tr>
<tr>
<td>onUnmount</td>
<td>(GameBase obj, SceneObject mountObj, int node)</td>
</tr>
</tbody>
</table>

**Types:**
- ShapeBaseData
- GameBaseData
- PlayerData
- SimObject
<table>
<thead>
<tr>
<th>Function</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>pickupRadius</td>
<td>PlayerData</td>
</tr>
<tr>
<td>proneBoundingBox</td>
<td>PlayerData</td>
</tr>
<tr>
<td>proneForce</td>
<td>PlayerData</td>
</tr>
<tr>
<td>recoverDelay</td>
<td>PlayerData</td>
</tr>
<tr>
<td>recoverRunForceScale</td>
<td>PlayerData</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>renderFirstPerson</td>
<td>PlayerData</td>
</tr>
<tr>
<td>renderWhenDestroyed</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>repairRate</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>runEnergyDrain</td>
<td>PlayerData</td>
</tr>
<tr>
<td>runForce</td>
<td>PlayerData</td>
</tr>
<tr>
<td>runSurfaceAngle</td>
<td>PlayerData</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>shadowEnable</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowMaxVisibleDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowProjectionDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSize</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSphereAdjust</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shapeFile</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shapeNameFP</td>
<td>PlayerData</td>
</tr>
<tr>
<td>Splash</td>
<td>PlayerData</td>
</tr>
<tr>
<td>splashAngle</td>
<td>PlayerData</td>
</tr>
<tr>
<td>splashEmitter</td>
<td>PlayerData</td>
</tr>
<tr>
<td>splashFreqMod</td>
<td>PlayerData</td>
</tr>
<tr>
<td>splashVelEpsilon</td>
<td>PlayerData</td>
</tr>
<tr>
<td>splashVelocity</td>
<td>PlayerData</td>
</tr>
<tr>
<td>sprintCanJump</td>
<td>PlayerData</td>
</tr>
<tr>
<td>sprintEnergyDrain</td>
<td>PlayerData</td>
</tr>
<tr>
<td>sprintForce</td>
<td>PlayerData</td>
</tr>
<tr>
<td>sprintPitchScale</td>
<td>PlayerData</td>
</tr>
<tr>
<td>sprintStrafeScale</td>
<td>PlayerData</td>
</tr>
<tr>
<td>sprintYawScale</td>
<td>PlayerData</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>swimBoundingBox</td>
<td>PlayerData</td>
</tr>
<tr>
<td>swimForce</td>
<td>PlayerData</td>
</tr>
<tr>
<td>transitionToLand</td>
<td>PlayerData</td>
</tr>
<tr>
<td>underwaterExplosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>upMaxSpeed</td>
<td>PlayerData</td>
</tr>
<tr>
<td>upResistFactor</td>
<td>PlayerData</td>
</tr>
<tr>
<td>Variable</td>
<td>Data Type</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>upResistSpeed</td>
<td>PlayerData</td>
</tr>
<tr>
<td>useEyePoint</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>waterBreathSound</td>
<td>PlayerData</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# PointLight Member List

This is the complete list of members for **PointLight**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>animate</td>
<td>LightBase</td>
</tr>
<tr>
<td>animationPeriod</td>
<td>LightBase</td>
</tr>
<tr>
<td>animationPhase</td>
<td>LightBase</td>
</tr>
<tr>
<td>animationType</td>
<td>LightBase</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>attenuationRatio</td>
<td>LightBase</td>
</tr>
<tr>
<td>brightness</td>
<td>LightBase</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>castShadows</td>
<td>LightBase</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>color</td>
<td>LightBase</td>
</tr>
<tr>
<td>cookie</td>
<td>LightBase</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>fadeStartDistance</td>
<td>LightBase</td>
</tr>
<tr>
<td>flareScale</td>
<td>LightBase</td>
</tr>
<tr>
<td>flareType</td>
<td>LightBase</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>includeLightmappedGeometryInShadow</td>
<td>LightBase</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEnabled</td>
<td>LightBase</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>isRenderable</td>
<td>PointLight</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>PointLight [static]</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>lastSplitTerrainOnly</td>
<td>LightBase</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>logWeight</td>
<td>LightObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>numSplits</td>
<td>LightBase</td>
</tr>
<tr>
<td>overDarkFactor</td>
<td>LightBase</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pauseAnimation</td>
<td>LightBase</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAnimation(LightAnimData anim)</td>
<td>LightBase</td>
</tr>
<tr>
<td>playAnimation()</td>
<td>LightBase</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>priority</td>
<td>LightBase</td>
</tr>
<tr>
<td>radius</td>
<td>PointLight</td>
</tr>
<tr>
<td>representedInLightmap</td>
<td>LightBase</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLightEnabled(bool state)</td>
<td>LightBase</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>shadowDarkenColor</td>
<td>LightBase</td>
</tr>
<tr>
<td>shadowDistance</td>
<td>LightBase</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>shadowSoftness</td>
<td>LightBase</td>
</tr>
<tr>
<td>shadowType</td>
<td>LightBase</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>texSize</td>
<td>LightBase</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td>SceneObject</td>
</tr>
</tbody>
</table>
## Portal Member List

This is the complete list of members for **Portal**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ambientLightColor</td>
<td>Zone</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>backSidePassable</td>
<td>Portal</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpZoneState(bool updateFirst=true)</td>
<td>Zone</td>
</tr>
<tr>
<td>edge</td>
<td>Zone</td>
</tr>
<tr>
<td>frontSidePassable</td>
<td>Portal</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getZoneId()</td>
<td>Zone</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExteriorPortal()</td>
<td>Portal</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInteriorPortal()</td>
<td>Portal</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>Portal</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>Portal</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>Method</td>
<td>Context</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(ScenObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>plane</td>
<td>Zone</td>
</tr>
<tr>
<td>point</td>
<td>Zone</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>soundAmbience</td>
<td>Zone</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>useAmbientLightColor</td>
<td>Zone</td>
</tr>
<tr>
<td>zoneGroup</td>
<td>Zone</td>
</tr>
</tbody>
</table>
# PostEffect Member List

This is the complete list of members for `PostEffect`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>acceptsAsChild</code> (SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>add</code> (SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>allowReflectPass</code></td>
<td>PostEffect</td>
</tr>
<tr>
<td><code>assignFieldsFrom</code> (SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>bringToFront</code> (SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>call</code> (string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>callOnChildren</code> (string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>callOnChildrenNoRecurse</code> (string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clear</code> ()</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>clearShaderMacros</code> ()</td>
<td>PostEffect</td>
</tr>
<tr>
<td><code>clone</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deleteAllObjects</code> ()</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>disable</code> ()</td>
<td>PostEffect</td>
</tr>
<tr>
<td><code>dump</code> (bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>dumpShaderDisassembly()</td>
<td>PostEffect</td>
</tr>
<tr>
<td>enable()</td>
<td>PostEffect</td>
</tr>
<tr>
<td><code>findObjectByInternalName</code></td>
<td>SimSet</td>
</tr>
<tr>
<td>(string internalName, bool searchChildren=false)</td>
<td></td>
</tr>
<tr>
<td>getAspectRatio()</td>
<td>PostEffect</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType</code> (string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue</code> (string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObject</code> (int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getObjectIndex</code> (SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEnabled</code></td>
<td><code>PostEffect</code></td>
</tr>
<tr>
<td><code>isEnabled()</code></td>
<td><code>PostEffect</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMember(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>listObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>onAdd()</code></td>
<td><code>PostEffect</code></td>
</tr>
<tr>
<td><code>onDisabled()</code></td>
<td><code>PostEffect</code></td>
</tr>
<tr>
<td><code>oneFrameOnly</code></td>
<td><code>PostEffect</code></td>
</tr>
<tr>
<td><code>onEnabled()</code></td>
<td><code>PostEffect</code></td>
</tr>
<tr>
<td><code>onObjectAdded(SimObject object)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>onObjectRemoved(SimObject object)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>onThisFrame</code></td>
<td><code>PostEffect</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>preProcess()</code></td>
<td><code>PostEffect</code></td>
</tr>
<tr>
<td><code>pushToBack(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>reload()</code></td>
<td><code>PostEffect</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>removeShaderMacro(string key)</td>
<td>PostEffect</td>
</tr>
<tr>
<td>renderBin</td>
<td>PostEffect</td>
</tr>
<tr>
<td>renderPriority</td>
<td>PostEffect</td>
</tr>
<tr>
<td>renderTime</td>
<td>PostEffect</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setShaderConst(string name, string value)</td>
<td>PostEffect</td>
</tr>
<tr>
<td>setShaderConsts()</td>
<td>PostEffect</td>
</tr>
<tr>
<td>setShaderMacro(string key, string value=&quot;&quot;)</td>
<td>PostEffect</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTexture(int index, string filePath)</td>
<td>PostEffect</td>
</tr>
<tr>
<td>shader</td>
<td>PostEffect</td>
</tr>
<tr>
<td>-------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>skip</td>
<td>PostEffect</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>stateBlock</td>
<td>PostEffect</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>target</td>
<td>PostEffect</td>
</tr>
<tr>
<td>targetClear</td>
<td>PostEffect</td>
</tr>
<tr>
<td>targetClearColor</td>
<td>PostEffect</td>
</tr>
<tr>
<td>targetDepthStencil</td>
<td>PostEffect</td>
</tr>
<tr>
<td>targetFormat</td>
<td>PostEffect</td>
</tr>
<tr>
<td>targetScale</td>
<td>PostEffect</td>
</tr>
<tr>
<td>targetSize</td>
<td>PostEffect</td>
</tr>
<tr>
<td>texture</td>
<td>PostEffect</td>
</tr>
<tr>
<td>toggle()</td>
<td>PostEffect</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# Precipitation Member List

This is the complete list of members for Precipitation, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>animateSplashes</td>
<td>Precipitation</td>
</tr>
<tr>
<td>applyImpulse(Point3F pos, VectorF vel)</td>
<td>GameBase</td>
</tr>
<tr>
<td>applyRadialImpulse(Point3F origin, float radius, float magnitude)</td>
<td>GameBase</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>boundingBox</td>
<td>GameBase [static]</td>
</tr>
<tr>
<td>boxHeight</td>
<td>Precipitation</td>
</tr>
<tr>
<td>boxWidth</td>
<td>Precipitation</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dataBlock</td>
<td>GameBase</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>doCollision</td>
<td>Precipitation</td>
</tr>
<tr>
<td>dropAnimateMS</td>
<td>Precipitation</td>
</tr>
<tr>
<td>dropSize</td>
<td>Precipitation</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
</tbody>
</table>
dumpGroupHierarchy()  SimObject
dumpMethods()  SimObject
fadeDist  Precipitation
fadeDistEnd  Precipitation
followCam  Precipitation
getCanSave()  SimObject
getClassName()  SimObject
getClassNamespace()  SimObject
getClientObject()  NetObject
dataBlock()  GameBase
dataDebugInfo()  SimObject
dataDeclarationLine()  SimObject
dataDynamicField(int index)  SimObject
dataDynamicFieldCount()  SimObject
dataEulerRotation()  SceneObject
dataField(int index)  SimObject
dataFieldCount()  SimObject
dataFieldType(string fieldName)  SimObject
dataFieldValue(string fieldName, int index=-1)  SimObject
dataFilename()  SimObject
dataForwardVector()  SceneObject
dataGhostID()  NetObject
dataGroup()  SimObject
dataId()  SimObject
dataInternalName()  SimObject
dataInverseTransform()  SceneObject
dataMountedObject(int slot)  SceneObject
<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScale()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>glowIntensity</code></td>
<td><code>Precipitation</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>hitPlayers</code></td>
<td><code>Precipitation</code></td>
</tr>
<tr>
<td><code>hitVehicles</code></td>
<td><code>Precipitation</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method/Attribute</td>
<td>Class/Type</td>
</tr>
<tr>
<td>------------------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>isMemberOfClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>Precipitation [static]</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>Precipitation [static]</td>
</tr>
<tr>
<td>isSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>maxMass</td>
<td>Precipitation</td>
</tr>
<tr>
<td>maxSpeed</td>
<td>Precipitation</td>
</tr>
<tr>
<td>maxTurbulence</td>
<td>Precipitation</td>
</tr>
<tr>
<td>minMass</td>
<td>Precipitation</td>
</tr>
<tr>
<td>minSpeed</td>
<td>Precipitation</td>
</tr>
<tr>
<td>modifyStorm(float percentage=1.0f, float seconds=5.0f)</td>
<td>Precipitation</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>numDrops</td>
<td>Precipitation</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Associated Type</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>reflect</td>
<td>Precipitation</td>
</tr>
<tr>
<td>rotateWithCamVel</td>
<td>Precipitation</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setControl</td>
<td>GameBase</td>
</tr>
<tr>
<td>setDataType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPercentage</td>
<td>Precipitation</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setTurbulence(float max=1.0f, float speed=5.0f, float seconds=5.0)</code></td>
<td><code>Precipitation</code></td>
</tr>
<tr>
<td><code>splashMS</code></td>
<td><code>Precipitation</code></td>
</tr>
<tr>
<td><code>splashSize</code></td>
<td><code>Precipitation</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>turbulenceSpeed</code></td>
<td><code>Precipitation</code></td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>useLighting</code></td>
<td><code>Precipitation</code></td>
</tr>
<tr>
<td><code>useTrueBillboards</code></td>
<td><code>Precipitation</code></td>
</tr>
<tr>
<td><code>useTurbulence</code></td>
<td><code>Precipitation</code></td>
</tr>
<tr>
<td><code>useWind</code></td>
<td><code>Precipitation</code></td>
</tr>
</tbody>
</table>
PrecipitationData Member List

This is the complete list of members for PrecipitationData, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>category</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dropShader</td>
<td>PrecipitationData</td>
</tr>
<tr>
<td>dropsPerSide</td>
<td>PrecipitationData</td>
</tr>
<tr>
<td>dropTexture</td>
<td>PrecipitationData</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAdd(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onMount(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onNewDataBlock(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td><code>onRemove</code> (GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td><code>onUnmount</code> (GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>reloadOnLocalClient()</code></td>
<td>SimDataBlock</td>
</tr>
<tr>
<td><code>save</code> (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule</code> (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCanSave</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>soundProfile</code></td>
<td>PrecipitationData</td>
</tr>
<tr>
<td><code>splashesPerSide</code></td>
<td>PrecipitationData</td>
</tr>
<tr>
<td><code>splashShader</code></td>
<td>PrecipitationData</td>
</tr>
<tr>
<td><code>splashTexture</code></td>
<td>PrecipitationData</td>
</tr>
</tbody>
</table>
## Prefab Member List

This is the complete list of members for Prefab, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>fileName</td>
<td>Prefab</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectName()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>Prefab</code> [static]</td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>Prefab</code> [static]</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>mountNode</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountPID</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountPos</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountRot</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method/Property</td>
<td>Class/Type</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onLoad(SimGroup children)</td>
<td>Prefab</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;))</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldNamespace(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>unmount</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountObject</code></td>
<td><code>SceneObject</code></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# Projectile Member List

This is the complete list of members for *Projectile*, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>applyImpulse(Point3F pos, VectorF vel)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>applyRadialImpulse(Point3F origin, float radius, float magnitude)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>boundingBox</code></td>
<td>GameBase [static]</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dataBlock</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>initialPosition</code></td>
<td><code>Projectile</code></td>
</tr>
<tr>
<td><code>initialVelocity</code></td>
<td><code>Projectile</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>Projectile</code> [static]</td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>Projectile</code> [static]</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>--------</td>
<td>-----------</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SceneObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>presimulate(float seconds=1.0f)</td>
<td>Projectile</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCansave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setControl(bool controlled)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setDataBlock(GameBaseData data)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>sourceObject</td>
<td>Projectile</td>
</tr>
<tr>
<td>sourceSlot</td>
<td>Projectile</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
</tbody>
</table>
# ProjectileData Member List

This is the complete list of members for **ProjectileData**, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>armingDelay</code></td>
<td><code>ProjectileData</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>bounceElasticity</code></td>
<td><code>ProjectileData</code></td>
</tr>
<tr>
<td><code>bounceFriction</code></td>
<td><code>ProjectileData</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>category</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>decal</code></td>
<td><code>ProjectileData</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>Explosion</code></td>
<td><code>ProjectileData</code></td>
</tr>
<tr>
<td><code>fadeDelay</code></td>
<td><code>ProjectileData</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>gravityMod</code></td>
<td><code>ProjectileData</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>impactForce</code></td>
<td><code>ProjectileData</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isBallistic</code></td>
<td><code>ProjectileData</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Variable/Method</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>lifetime</td>
<td>ProjectileData</td>
</tr>
<tr>
<td>lightDesc</td>
<td>ProjectileData</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>muzzleVelocity</td>
<td>ProjectileData</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAdd(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onCollision(Projectile proj, SceneObject col, float fade, Point3F pos, Point3F normal)</td>
<td>ProjectileData</td>
</tr>
<tr>
<td>onExplode(Projectile proj, Point3F pos, float fade)</td>
<td>ProjectileData</td>
</tr>
<tr>
<td>onMount(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onNewDataBlock(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onRemove(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onUnmount(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>ParticleEmitter</td>
<td>ProjectileData</td>
</tr>
<tr>
<td>particleWaterEmitter</td>
<td>ProjectileData</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>projectileShapeName</td>
<td>ProjectileData</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>ProjectileData</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code> (string newNameName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>sound</code></td>
<td><code>ProjectileData</code></td>
</tr>
<tr>
<td><code>Splash</code></td>
<td><code>ProjectileData</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>velInheritFactor</code></td>
<td><code>ProjectileData</code></td>
</tr>
<tr>
<td><code>waterExplosion</code></td>
<td><code>ProjectileData</code></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## ProximityMine Member List

This is the complete list of members for **ProximityMine**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>applyDamage(float amount)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>applyImpulse(Point3F pos, Point3F vec)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>GameBase::applyImpulse(Point3F pos, VectorF vel)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>applyRadialImpulse(Point3F origin, float radius, float magnitude)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>applyRepair(float amount)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>boundingBox</code></td>
<td><code>GameBase</code> [static]</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canCloak()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>changeMaterial(string mapTo, Material oldMat, Material newMat)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dataBlock</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>destroyThread(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMeshVisibility()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>explode()</code></td>
<td>ProximityMine</td>
</tr>
<tr>
<td><code>getAIRepairPoint()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getCameraFov()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getControllingClient()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getControllingObject()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getDamageFlash()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getDamageLevel()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getDamagePercent()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getDamageState()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getDataBlock()</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDefaultCameraFov()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getEnergyLevel()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getEnergyPercent()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>getEulerRotation()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getEyePoint()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td></td>
<td>Class</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>getEyeTransform()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeVector()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getImageAltTrigger(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageAmmo(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageGenericTrigger(int slot, int trigger)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageLoaded(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageScriptAnimPrefix(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageSkinTag(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageState(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageTarget(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageTrigger(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getLastStickyNormal()</td>
<td>Item</td>
</tr>
<tr>
<td>getLastStickyPos()</td>
<td>Item</td>
</tr>
<tr>
<td>getLookAtPoint(float distance=2000, int typeMask=0xFFFFFFFF)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getMaxDamage()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getModelFile()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountSlot(ShapeBaseImageData)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzlePoint(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzleVector(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPendingImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getPosition(int slot)</td>
<td>SceneBase</td>
</tr>
<tr>
<td>getRechargeRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRepairRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getShapeName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSkinName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSlotTransform(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTargetCount()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getNameTarget(int index)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getVelocity()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getWhiteOut()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hasImageState(int slot, string state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isAIControlled</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isAtRest()</td>
<td>Item</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isCloaked()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isDestroyed()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isDisabled()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEnabled()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isHidden()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isImageFiring(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isImageMounted(ShapeBaseImageData image)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>ProximityMine</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isRotating()</td>
<td>Item</td>
</tr>
<tr>
<td>isSelecteable</td>
<td>ProximityMine</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isStatic()</td>
<td>Item</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>maxWarpTicks</td>
<td>Item</td>
</tr>
<tr>
<td>minWarpTicks</td>
<td>Item</td>
</tr>
<tr>
<td>mountImage(ShapeBaseImageData image, int slot, bool loaded=true, string skinTag=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onEnterLiquid(string objID, string waterCoverage, string liquidType)</td>
<td>Item</td>
</tr>
<tr>
<td>onLeaveLiquid(string objID, string liquidType)</td>
<td>Item</td>
</tr>
<tr>
<td>onStickyCollision(string objID)</td>
<td>Item</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>pauseThread</code> (int slot)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>playAudio</code> (int slot, SFXTrack track)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>playThread</code> (int slot, string name=&quot;&quot;)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>position</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>rotate</code></td>
<td><code>Item</code></td>
</tr>
<tr>
<td><code>rotation</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>save</code> (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>scale</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>schedule</code> (float time, string method, string args...)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>scopeToClient</code> (NetConnection client)</td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setAllMeshesHidden</code> (bool hide)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setCameraFov</code> (float fov)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setCanSave</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCloaked</code> (bool cloak)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setCollisionTimeout</code> (int ignoreColObj=NULL)</td>
<td><code>Item</code></td>
</tr>
<tr>
<td><code>setControl</code> (bool controlled)</td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>setDamageFlash</code> (float level)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setDamageLevel</code> (float level)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setDamageState</code> (string state)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setDamageVector</code> (Point3F vec)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setDataBlock</code> (GameBaseData data)</td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEnergyLevel</code> (float level)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool show)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAltTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAmmo(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageGenericTrigger(int slot, int trigger, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageLoaded(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageScriptAnimPrefix(int slot, string prefix)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTarget(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInvincibleMode(float time, float speed)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setMeshHidden(string name, bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setRechargeRate(float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setRepairRate(float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setShapeName(string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setSkinName(string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setThreadDir</code> (int slot, bool fwd)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setThreadPosition</code> (int slot, float pos)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setThreadTimeScale</code> (int slot, float scale)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setTransform</code> (TransformF txfm)</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setVelocity</code> (Point3F vel)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setWhiteOut</code> (float level)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>skin</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>startFade</code> (int time, int delay, bool fadeOut)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>static</code></td>
<td><code>Item</code></td>
</tr>
<tr>
<td><code>stopAudio</code> (int slot)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>stopThread</code> (int slot)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>unmount</code> ()</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountImage</code> (int slot)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>unmountObject</code> (SceneObject target)</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>validateCameraFov</code> (float fov)</td>
<td><code>ShapeBase</code></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# ProximityMineData Member List

This is the complete list of members for `ProximityMineData`, including all inherited members.

<table>
<thead>
<tr>
<th>Method/Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>armingDelay</td>
<td>ProximityMineData</td>
</tr>
<tr>
<td>armingSound</td>
<td>ProximityMineData</td>
</tr>
<tr>
<td>assignFieldsFrom</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>autoTriggerDelay</td>
<td>ProximityMineData</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>cameraDefaultFov</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cameraMaxDist</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cameraMaxFov</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cameraMinDist</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cameraMinFov</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>category</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>checkDeployPos(TransformF txfm)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>computeCRC</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cubeReflectorDesc</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>Debris</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>debrisShapeName</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>density</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>destroyedLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>disabledLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>drag</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><strong>dump</strong>(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>elasticity</td>
<td>ItemData</td>
</tr>
<tr>
<td>Explosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>explosionOffset</td>
<td>ProximityMineData</td>
</tr>
<tr>
<td>firstPersonOnly</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>friction</td>
<td>ItemData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeployTransform**(Point3F pos, Point3F normal)**</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>getDynamicField**(int index)**</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField**(int index)**</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType**(string fieldName)**</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue**(string fieldName, int index=-1)**</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>gravityMod</code></td>
<td>ItemData</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>inheritEnergyFromMount</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInvincible</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>lightColor</code></td>
<td>ItemData</td>
</tr>
<tr>
<td><code>lightOnlyStatic</code></td>
<td>ItemData</td>
</tr>
<tr>
<td><code>lightRadius</code></td>
<td>ItemData</td>
</tr>
<tr>
<td><code>lightTime</code></td>
<td>ItemData</td>
</tr>
<tr>
<td><code>lightType</code></td>
<td>ItemData</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>mass</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>maxDamage</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>maxEnergy</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>maxVelocity</code></td>
<td>ItemData</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td><strong>name</strong></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><strong>observeThroughObject</strong></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><strong>onAdd</strong> <code>(GameBase obj)</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><strong>onCollision</strong> <code>(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><strong>onDamage</strong> <code>(ShapeBase obj, float delta)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><strong>onDestroyed</strong> <code>(ShapeBase obj, string lastState)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><strong>onDisabled</strong> <code>(ShapeBase obj, string lastState)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><strong>onEnabled</strong> <code>(ShapeBase obj, string lastState)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><strong>onEndSequence</strong> <code>(ShapeBase obj, int slot)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><strong>onExplode</strong> <code>(ProximityMine obj, Point3F pos)</code></td>
<td><code>ProximityMineData</code></td>
</tr>
<tr>
<td><strong>onForceUncloak</strong> <code>(ShapeBase obj, string reason)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><strong>onImpact</strong> <code>(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><strong>onMount</strong> <code>(GameBase obj, SceneObject mountObj, int node)</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><strong>onNewDataBlock</strong> <code>(GameBase obj)</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><strong>onRemove</strong> <code>(GameBase obj)</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><strong>onTrigger</strong> <code>(ShapeBase obj, int index, bool state)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><strong>onTriggered</strong> <code>(ProximityMine obj, SceneObject target)</code></td>
<td><code>ProximityMineData</code></td>
</tr>
<tr>
<td><strong>onUnmount</strong> <code>(GameBase obj, SceneObject mountObj, int node)</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><strong>parentGroup</strong></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><strong>persistentId</strong></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><strong>reloadOnLocalClient</strong></td>
<td><code>SimDataBlock</code></td>
</tr>
<tr>
<td><strong>renderWhenDestroyed</strong></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>repairRate</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>shadowEnable</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowMaxVisibleDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowProjectionDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSize</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSphereAdjust</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shapeFile</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>simpleServerCollision</td>
<td>ItemData</td>
</tr>
<tr>
<td>sticky</td>
<td>ItemData</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>Variable</td>
<td>Data Type</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>triggerDelay</td>
<td>ProximityMineData</td>
</tr>
<tr>
<td>triggerOnOwner</td>
<td>ProximityMineData</td>
</tr>
<tr>
<td>triggerRadius</td>
<td>ProximityMineData</td>
</tr>
<tr>
<td>triggerSound</td>
<td>ProximityMineData</td>
</tr>
<tr>
<td>triggerSpeed</td>
<td>ProximityMineData</td>
</tr>
<tr>
<td>underwaterExplosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>useEyePoint</td>
<td>ShapeBaseData</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# PxCloth Member List

This is the complete list of members for **PxCloth**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>applyImpulse</code>(Point3F pos, VectorF vel)</td>
<td>GameBase</td>
</tr>
<tr>
<td><code>applyRadialImpulse</code>(Point3F origin, float radius, float magnitude)</td>
<td>GameBase</td>
</tr>
<tr>
<td><code>assignFieldsFrom</code>(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>attachments</code></td>
<td>PxCloth</td>
</tr>
<tr>
<td><code>bending</code></td>
<td>PxCloth</td>
</tr>
<tr>
<td><code>bendingStiffness</code></td>
<td>PxCloth</td>
</tr>
<tr>
<td><code>boundingBox</code></td>
<td>GameBase [static]</td>
</tr>
<tr>
<td><code>call</code>(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient</code>(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>damping</code></td>
<td>PxCloth</td>
</tr>
<tr>
<td><code>dampingCoefficient</code></td>
<td>PxCloth</td>
</tr>
<tr>
<td><code>dataBlock</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>deepClone</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>density</code></td>
<td>PxCloth</td>
</tr>
<tr>
<td><code>dump</code>(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>friction</td>
<td>PxCloth</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td><code>getMountNodeObject</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObjectBox</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getObjectMount</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getPosition</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getRightVector</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getScale</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getServerObject</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTransform</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>PxCloth</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>PxCloth</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>Material</td>
<td>PxCloth</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot,</td>
<td>SceneObject</td>
</tr>
<tr>
<td>TransformF txfm=MatrixF::Identity)</td>
<td></td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>samples</td>
<td>PxCloth</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>selfCollision</td>
<td>PxCloth</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td><code>setControl(bool controlled)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>setDataBlock(GameBaseData data)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>size</code></td>
<td><code>PxCloth</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>thickness</code></td>
<td><code>PxCloth</code></td>
</tr>
<tr>
<td><code>triangleCollision</code></td>
<td><code>PxCloth</code></td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td><code>SceneObject</code></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## PxMaterial Member List

This is the complete list of members for **PxMaterial**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dynamicFriction</td>
<td>PxMaterial</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>reloadOnLocalClient()</code></td>
<td>SimDataBlock</td>
</tr>
<tr>
<td><code>restitution</code></td>
<td>PxMaterial</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class Name</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>staticFriction</td>
<td>PxMaterial</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>
# PxBert Actor Member List

This is the complete list of members for PxBert Actor, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applyImpulse(Point3F pos, VectorF vel)</td>
<td>GameBase</td>
</tr>
<tr>
<td>applyRadialImpulse(Point3F origin, float radius, float magnitude)</td>
<td>GameBase</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>boundingBox</td>
<td>GameBase [static]</td>
</tr>
<tr>
<td>broken</td>
<td>PxBert Actor</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dataBlock</td>
<td>GameBase</td>
</tr>
<tr>
<td>debugRender</td>
<td>PxBert Actor</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>PxMultiActor [static]</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>PxMultiActor [static]</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>listMeshes</td>
<td>(enum Hidden/Shown/All)</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save</td>
<td>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule</td>
<td>(float time, string method, string args...)</td>
</tr>
<tr>
<td>scopeToClient</td>
<td>(NetConnection client)</td>
</tr>
<tr>
<td>setAllHidden</td>
<td>(bool)</td>
</tr>
<tr>
<td>setBroken</td>
<td>(bool)</td>
</tr>
<tr>
<td>setCanSave</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>(string name)</td>
</tr>
<tr>
<td>setControl</td>
<td>(bool controlled)</td>
</tr>
<tr>
<td>setDataBlock</td>
<td>(GameBaseData data)</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setFieldType</td>
<td>(string fieldName, string type)</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>(string fieldName, string)</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setMeshHidden</code></td>
<td><code>PxMultiActor</code></td>
</tr>
<tr>
<td><code>setName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setScale</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>unmount</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountObject</code></td>
<td><code>SceneObject</code></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
**PxMultiActorData Member List**

This is the complete list of members for `PxMultiActorData`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>angularDrag</code></td>
<td><code>PxMultiActorData</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>breakForce</code></td>
<td><code>PxMultiActorData</code></td>
</tr>
<tr>
<td><code>buoyancyDensity</code></td>
<td><code>PxMultiActorData</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>category</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clientOnly</code></td>
<td><code>PxMultiActorData</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpModel</code></td>
<td><code>PxMultiActorData</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>linearDrag</code></td>
<td><code>PxMultiActorData</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>Material</code></td>
<td><code>PxMultiActorData</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>noCorrection</code></td>
<td><code>PxMultiActorData</code></td>
</tr>
<tr>
<td><code>onAdd(GameBase obj)</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onMount(GameBase obj, SceneObject mountObj, int node)</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onNewDataBlock(GameBase obj)</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onRemove(GameBase obj)</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onUnmount(GameBase obj, SceneObject mountObj, int node)</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>physXStream</code></td>
<td><code>PxMultiActorData</code></td>
</tr>
<tr>
<td><code>reload</code></td>
<td><code>PxMultiActorData</code></td>
</tr>
<tr>
<td><code>reloadOnLocalClient()</code></td>
<td><code>SimDataBlock</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;\&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>shapeName</code></td>
<td>PxMultiActorData</td>
</tr>
<tr>
<td><code>singlePlayerOnly</code></td>
<td>PxMultiActorData</td>
</tr>
<tr>
<td><code>string</code></td>
<td>PxMultiActorData</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>waterDragScale</code></td>
<td>PxMultiActorData</td>
</tr>
</tbody>
</table>
RadialImpulseEvent Member List

This is the complete list of members for RadialImpulseEvent, including all inherited members.

```
send(string inPosition="1.0 1.0 1.0",
    float radius=10.0f, float
    magnitude=20.0f)  RadialImpulseEvent [static]
```

Copyright © GarageGames, LLC. All Rights Reserved.
ReflectorDesc Member List

This is the complete list of members for **ReflectorDesc**, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>detailAdjust</td>
<td>ReflectorDesc</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>farDist</td>
<td>ReflectorDesc</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>maxRateMs</td>
<td>ReflectorDesc</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>nearDist</td>
<td>ReflectorDesc</td>
</tr>
<tr>
<td>objectTypeMask</td>
<td>ReflectorDesc</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>priority</td>
<td>ReflectorDesc</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td><strong>save</strong>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>schedule</strong>(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setCanSave</strong>(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setClassNamespace</strong>(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setEditorOnly</strong>(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setFieldType</strong>(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setFieldValue</strong>(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setFilename</strong>(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setHidden</strong>(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setInternalName</strong>(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setIsExpanded</strong>(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setIsSelected</strong>(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setLocked</strong>(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setName</strong>(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setNameChangeAllowed</strong>(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setSuperClassNamespace</strong>(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>superClass</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>texSize</strong></td>
<td>ReflectorDesc</td>
</tr>
<tr>
<td><strong>useOcclusionQuery</strong></td>
<td>ReflectorDesc</td>
</tr>
</tbody>
</table>
# RenderBinManager Member List

This is the complete list of members for `RenderBinManager`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom <code>SimObject fromObject</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>assignPersistentId ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>binType</code></td>
<td><code>RenderBinManager</code></td>
</tr>
<tr>
<td><code>call</code> <code>string method, string args...</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump</code> <code>bool detailed=false</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getBinType</code> ()</td>
<td><code>RenderBinManager</code></td>
</tr>
<tr>
<td><code>getCanSave</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField</code> <code>int index</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getField</code> <code>int index</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>processAddOrder</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>renderOrder</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>schedule</code></td>
<td><code>float time, string method, string args...</code></td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td><code>bool value=true</code></td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td><code>string name</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td><code>bool value=true</code></td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td><code>string fieldName, string type</code></td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td><code>string fieldName, string value, int index=-1</code></td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td><code>string fileName</code></td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td><code>bool value=true</code></td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td><code>string newInternalName</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td><code>bool state=true</code></td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td><code>bool state=true</code></td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td><code>bool value=true</code></td>
</tr>
<tr>
<td><code>setName</code></td>
<td><code>string newName</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td><code>bool value=true</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>string name</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
### RenderFormatToken Member List

This is the complete list of members for `RenderFormatToken`, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>aaLevel</td>
<td><code>RenderFormatToken</code></td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>canSave</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>class</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>className</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>clone()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>copyEffect</td>
<td><code>RenderFormatToken</code></td>
</tr>
<tr>
<td>deepClone()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>delete()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>depthFormat</td>
<td><code>RenderFormatToken</code></td>
</tr>
<tr>
<td>disable()</td>
<td><code>RenderPassStateToken</code></td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpMethods()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>enable()</td>
<td><code>RenderPassStateToken</code></td>
</tr>
<tr>
<td>enabled</td>
<td><code>RenderPassStateToken</code></td>
</tr>
<tr>
<td>format</td>
<td><code>RenderFormatToken</code></td>
</tr>
<tr>
<td>getCanSave()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>resolveEffect</td>
<td>RenderFormatToken</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>toggle</td>
<td>RenderPassStateToken</td>
</tr>
</tbody>
</table>
# RenderGlowMgr Member List

This is the complete list of members for `RenderGlowMgr`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(simObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>binType</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getBinType()</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>processAddOrder</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>renderOrder</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td><code>schedule</code></td>
<td><code>schedule</code> takes three arguments: <code>time</code>, <code>method</code>, and <code>args...</code></td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td>Sets the <code>canSave</code> property to <code>true</code> or <code>false</code></td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td>Sets the <code>classNamespace</code> property to <code>name</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td>Sets the <code>editorOnly</code> property to <code>true</code> or <code>false</code></td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td>Sets the <code>fieldType</code> property to <code>fieldName</code> and <code>type</code></td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td>Sets the <code>fieldValue</code> property to <code>fieldName</code>, <code>value</code>, and <code>index=-1</code></td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td>Sets the <code>filename</code> property to <code>fileName</code></td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td>Sets the <code>hidden</code> property to <code>true</code> or <code>false</code></td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td>Sets the <code>internalName</code> property to <code>newInternalName</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td>Sets the <code>isExpanded</code> property to <code>true</code> or <code>false</code></td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td>Sets the <code>isSelected</code> property to <code>true</code> or <code>false</code></td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td>Sets the <code>locked</code> property to <code>true</code> or <code>false</code></td>
</tr>
<tr>
<td><code>setName</code></td>
<td>Sets the <code>name</code> property to <code>newName</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td>Sets the <code>supportedNameChange</code> property to <code>true</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td>Sets the <code>superClassNamespace</code> property to <code>name</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>Represents the <code>superClass</code> property</td>
</tr>
</tbody>
</table>
### RenderImposterMgr Member List

This is the complete list of members for `RenderImposterMgr`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code> (SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>binType</code></td>
<td>RenderBinManager</td>
</tr>
<tr>
<td><code>call</code> (string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clone</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump</code> (bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getBinType</code> ()</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td><code>getCanSave</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField</code> (int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField</code> (int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>processAddOrder</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>renderOrder</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>schedule</td>
<td>float time, string method, string args...</td>
</tr>
<tr>
<td>setCanSave</td>
<td>bool value=true</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>string name</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>bool value=true</td>
</tr>
<tr>
<td>setFieldType</td>
<td>string fieldName, string type</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>string fieldName, string value, int index=-1</td>
</tr>
<tr>
<td>setFilename</td>
<td>string fileName</td>
</tr>
<tr>
<td>setHidden</td>
<td>bool value=true</td>
</tr>
<tr>
<td>setInternalName</td>
<td>string newInternalName</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>bool state=true</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>bool state=true</td>
</tr>
<tr>
<td>setLocked</td>
<td>bool value=true</td>
</tr>
<tr>
<td>setName</td>
<td>string newName</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>bool value=true</td>
</tr>
<tr>
<td>setSuperClassNamespace</td>
<td>string name</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# RenderMeshExample Member List

This is the complete list of members for `RenderMeshExample`, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>RenderMeshExample [static]</code></td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>RenderMeshExample [static]</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>Material</code></td>
<td><code>RenderMeshExample</code></td>
</tr>
<tr>
<td><code>mountNode</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountObject(SceneObject objB, int slot, TransformF)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>txfm=MatrixF::Identity</td>
<td></td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>postApply()</td>
<td>RenderMeshExample</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
</tbody>
</table>
# RenderMeshMgr Member List

This is the complete list of members for `RenderMeshMgr`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>binType</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getBinType()</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>processAddOrder RenderBinManager</code></td>
<td>RenderBinManager</td>
</tr>
<tr>
<td><code>renderOrder RenderBinManager</code></td>
<td>RenderBinManager</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td><code>schedule</code></td>
<td>float time, string method, string args...</td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td>bool value=true</td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td>string name</td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td>bool value=true</td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td>string fieldName, string type</td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td>string fieldName, string value, int index=-1</td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td>string fileName</td>
</tr>
<tr>
<td><code> setHidden</code></td>
<td>bool value=true</td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td>string newName</td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td>bool state=true</td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td>bool state=true</td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td>bool value=true</td>
</tr>
<tr>
<td><code>setName</code></td>
<td>string newName</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td>bool value=true</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td>string name</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
</tbody>
</table>
**RenderObjectExample Member List**

This is the complete list of members for `RenderObjectExample`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>RenderObjectExample</code> [static]</td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>RenderObjectExample</code> [static]</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>mountNode</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked</td>
<td>SimObject</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td><code>SceneObject</code></td>
</tr>
</tbody>
</table>
# RenderObjectMgr Member List

This is the complete list of members for `RenderObjectMgr`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>binType</code></td>
<td><code>RenderBinManager</code></td>
</tr>
<tr>
<td><code>call</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getBinType</code></td>
<td><code>RenderBinManager</code></td>
</tr>
<tr>
<td><code>getCanSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>processAddOrder</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>renderOrder</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>schedule(float time, string method, string</td>
<td>Schedule function</td>
</tr>
<tr>
<td>args...)</td>
<td></td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>Set whether object can be saved</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>Set the class namespace</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>Set whether object is only for editors</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>Set the field type</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value,</td>
<td>Set the field value</td>
</tr>
<tr>
<td>int index=-1)</td>
<td></td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>Set the file name</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>Set whether object is hidden</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>Set the internal name</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>Set whether object is expanded</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>Set whether object is selected</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>Set whether object is locked</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>Set the name</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>Set whether name change is allowed</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>Set the super class namespace</td>
</tr>
<tr>
<td>superClass</td>
<td></td>
</tr>
</tbody>
</table>
## RenderOcclusionMgr Member List

This is the complete list of members for `RenderOcclusionMgr`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>binType</code></td>
<td><code>RenderBinManager</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>debugRender</code></td>
<td><code>RenderOcclusionMgr [static]</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getBinType()</code></td>
<td><code>RenderBinManager</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>processAddOrder</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>renderOrder</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td><code>save</code> (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule</code> (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCanSave</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
</tbody>
</table>
**RenderParticleMgr Member List**

This is the complete list of members for `RenderParticleMgr`, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>binType</code></td>
<td><code>RenderBinManager</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getBinType()</code></td>
<td><code>RenderBinManager</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden()</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked()</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>processAddOrder</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>renderOrder</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>schedule</td>
<td>(float time, string method, string args...)</td>
</tr>
<tr>
<td>setCanSave</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>(string name)</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setFieldType</td>
<td>(string fieldName, string type)</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>(string fieldName, string value, int index=-1)</td>
</tr>
<tr>
<td>setFilename</td>
<td>(string fileName)</td>
</tr>
<tr>
<td>setHidden</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setInternalName</td>
<td>(string newInternalName)</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>(bool state=true)</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>(bool state=true)</td>
</tr>
<tr>
<td>setLocked</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setName</td>
<td>(string newName)</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setSuperClassNamespace</td>
<td>(string name)</td>
</tr>
<tr>
<td>superClass</td>
<td></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
**RenderPassManager Member List**

This is the complete list of members for `RenderPassManager`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>addManager</code> (RenderBinManager renderBin)</td>
<td><code>RenderPassManager</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom</code> (SimObject fromObject)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call</code> (string method, string args...)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump</code> (bool detailed=false)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCanSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField</code> (int index)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getField</code> (int index)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td><code>getFieldType</code> (string fieldName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue</code> (string fieldName, int index=-1)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getManager</code> (int index)</td>
<td><code>RenderPassManager</code></td>
</tr>
<tr>
<td><code>getManagerCount</code></td>
<td><code>RenderPassManager</code></td>
</tr>
<tr>
<td><code>getName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup</code> (SimGroup group)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField</code> (string fieldName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass</code> (string className)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod</code> (string methodName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>removeManager</code> (RenderBinManager renderBin)</td>
<td><code>RenderPassManager</code></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td><code>save(fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule(time, method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace(name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType(fieldName, type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue(fieldName, value, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename(fileName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName(newInternalName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded(state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected(state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
### RenderPassStateBin Member List

This is the complete list of members for RenderPassStateBin, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>binType</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getBinType()</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>processAddOrder(RenderBinManager)</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>renderOrder</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td><code>schedule</code> (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCanSave</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>stateToken</code></td>
<td>RenderPassStateBin</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
</tbody>
</table>
# RenderPassStateToken Member List

This is the complete list of members for `RenderPassStateToken`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>disable()</code></td>
<td><code>RenderPassStateToken</code></td>
</tr>
<tr>
<td><code>dump</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>enable()</code></td>
<td><code>RenderPassStateToken</code></td>
</tr>
<tr>
<td><code>enabled</code></td>
<td><code>RenderPassStateToken</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
</tbody>
</table>
getField(int index) SimObject
getFieldCount() SimObject
getFieldType(string fieldName) SimObject
getFieldValue(string fieldName, int index=-1) SimObject
getFilename() SimObject
getGroup() SimObject
getld() SimObject
getInternalName() SimObject
getName() SimObject
getSuperClassNamespace() SimObject
hidden SimObject
internalName SimObject
isChildOfGroup(SimGroup group) SimObject
isEditorOnly() SimObject
isExpanded() SimObject
isField(string fieldName) SimObject
isInNamespaceHierarchy(string name) SimObject
isMemberOfClass(string className) SimObject
isMethod(string methodName) SimObject
isNameChangeAllowed() SimObject
isSelected() SimObject
locked SimObject
name SimObject
parentGroup SimObject
persistentId SimObject
save(string fileName, bool selectedOnly=false, string SimObject
<table>
<thead>
<tr>
<th>Method Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>schedule</td>
<td>(float time, string method, string args...) SimObject</td>
</tr>
<tr>
<td>setCanSave</td>
<td>(bool value=true) SimObject</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>(string name) SimObject</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>(bool value=true) SimObject</td>
</tr>
<tr>
<td>setFieldType</td>
<td>(string fieldName, string type) SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>(string fieldName, string value, int index=-1) SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>(string fileName) SimObject</td>
</tr>
<tr>
<td>setHidden</td>
<td>(bool value=true) SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>(string newInternalName) SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>(bool state=true) SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>(bool state=true) SimObject</td>
</tr>
<tr>
<td>setLocked</td>
<td>(bool value=true) SimObject</td>
</tr>
<tr>
<td>setName</td>
<td>(string newName) SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>(bool value=true) SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace</td>
<td>(string name) SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>toggle</td>
<td>RenderPassStateToken</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## RenderPrePassMgr Member List

This is the complete list of members for `RenderPrePassMgr`, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>binType</code></td>
<td><code>RenderBinManager</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getBinType()</code></td>
<td><code>RenderBinManager</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>processAddOrder</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>renderOrder</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>schedule</td>
<td>schedule(float time, string method, string args...) SimObject</td>
</tr>
<tr>
<td>setCanSave</td>
<td>setCanSave(bool value=true) SimObject</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>setClassNamespace(string name) SimObject</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>setEditorOnly(bool value=true) SimObject</td>
</tr>
<tr>
<td>setFieldType</td>
<td>setFieldType(string fieldName, string type) SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>setFieldValue(string fieldName, string value, int index=-1) SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>setFilename(string fileName) SimObject</td>
</tr>
<tr>
<td>setHidden</td>
<td>setHidden(bool value=true) SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>setInternalName(string newInternalName) SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>setIsExpanded(bool state=true) SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>setIsSelected(bool state=true) SimObject</td>
</tr>
<tr>
<td>setLocked</td>
<td>setLocked(bool value=true) SimObject</td>
</tr>
<tr>
<td>setName</td>
<td>setName(string newName) SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>setNameChangeAllowed(bool value=true) SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace</td>
<td>setSuperClassNamespace(string name) SimObject</td>
</tr>
</tbody>
</table>

superClass
## RenderShapeExample Member List

This is the complete list of members for *RenderShapeExample*, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Template</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>setUpVector</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>RenderShapeExample     [static]</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>RenderShapeExample     [static]</td>
</tr>
<tr>
<td>isSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot; )</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args…)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setScale</code> (Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setScopeAlways</code>()</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setTransform</code> (TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>shapeFile</code></td>
<td>RenderShapeExample</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>unmount</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>unmountObject</code> (SceneObject target)</td>
<td>SceneObject</td>
</tr>
</tbody>
</table>
## RenderTerrainMgr Member List

This is the complete list of members for `RenderTerrainMgr`, including all inherited members.

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>binType</code></td>
<td><code>RenderBinManager</code></td>
</tr>
<tr>
<td><code>call</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getBinType</code></td>
<td><code>RenderBinManager</code></td>
</tr>
<tr>
<td><code>getCanSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getld()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>processAddOrder(RenderBinManager)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>renderOrder</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>renderWireframe</td>
<td>RenderTerrainMgr [static]</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>
## RenderTexTargetBinManager Member List

This is the complete list of members for `RenderTexTargetBinManager`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>binType</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getBinType()</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>processAddOrder(RenderBinManager)</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>renderOrder(RenderBinManager)</td>
<td>RenderBinManager</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>schedule</code></td>
<td>(float time, string method, string args...)</td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td>(string name)</td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td>(string fieldName, string type)</td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td>(string fieldName, string value, int index=-1)</td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td>(string fileName)</td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td>(string newInternalName)</td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td>(bool state=true)</td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td>(bool state=true)</td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setName</code></td>
<td>(string newName)</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td>(string name)</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## RenderTranslucentMgr Member List

This is the complete list of members for `RenderTranslucentMgr`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code> <em>(SimObject fromObject)</em></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>binType</code></td>
<td><code>RenderBinManager</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getBinType()</code></td>
<td><code>RenderBinManager</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>processAddOrder</code></td>
<td><code>RenderBinManager</code></td>
</tr>
<tr>
<td><code>renderOrder</code></td>
<td><code>RenderBinManager</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>schedule</td>
<td>Schedule time, method, and arguments...</td>
</tr>
<tr>
<td>setCanSave</td>
<td>Set whether the object can be saved</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>Set the class namespace</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>Set whether the object is editor-only</td>
</tr>
<tr>
<td>setFieldType</td>
<td>Set the field type</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>Set the field value</td>
</tr>
<tr>
<td>setFilename</td>
<td>Set the filename</td>
</tr>
<tr>
<td>setHidden</td>
<td>Set whether the object is hidden</td>
</tr>
<tr>
<td>setInternalName</td>
<td>Set the internal name</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>Set whether the object is expanded</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>Set whether the object is selected</td>
</tr>
<tr>
<td>setLocked</td>
<td>Set whether the object is locked</td>
</tr>
<tr>
<td>setName</td>
<td>Set the name</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>Set whether the name change is allowed</td>
</tr>
<tr>
<td>setSuperClassNamespace</td>
<td>Set the super class namespace</td>
</tr>
<tr>
<td>superClass</td>
<td>Get the superclass</td>
</tr>
</tbody>
</table>
## RigidShape Member List

This is the complete list of members for **RigidShape**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>applyDamage(float amount)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>applyImpulse(Point3F pos, Point3F vec)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>GameBase::applyImpulse(Point3F pos, VectorF vel)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>applyRadialImpulse(Point3F origin, float radius, float magnitude)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>applyRepair(float amount)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>boundingBox</code></td>
<td>GameBase [static]</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canCloak()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>changeMaterial(string mapTo, Material oldMat, Material newMat)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dataBlock</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>destroyThread(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMeshVisibility()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>freezeSim(bool isFrozen)</code></td>
<td><code>RigidShape</code></td>
</tr>
<tr>
<td><code>getAIRepairPoint()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getCameraFov()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getControllingClient()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getControllingObject()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamageFlash()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamageLevel()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamagePercent()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamageState()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDataBlock()</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDefaultCameraFov()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getEnergyLevel()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getEnergyPercent()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getEulerRotation()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getEyePoint()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getEyeTransform()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Function</td>
<td>Returns</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>getEyeVector()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getImageAltTrigger(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageAmmo(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageGenericTrigger(int slot, int trigger)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageLoaded(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageScriptAnimPrefix(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageSkinTag(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageState(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageTarget(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageTrigger(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getLookAtPoint(float distance=2000, int typeMask=0xFFFFFFFF)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMaxDamage()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getModelFile()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountSlot(ShapeBaseImageData)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMuzzlePoint(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMuzzleVector(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getPendingImage(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getRechargeRate()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getRepairRate()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScale()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getShapeName()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getSkinName()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getSlotTransform(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTargetCount()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getTargetName(int index)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getVelocity()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getWhiteOut()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>hasImageState(int slot, string state)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isAIControlled</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>isCloaked()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isDestroyed()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isDisabled()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEnabled()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isHidden()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isImageFiring(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isImageMounted(ShapeBaseImageData image)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td>RigidShape [static]</td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td>RigidShape [static]</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountImage(ShapeBaseImageData image, int slot, bool loaded=true, string skinTag=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SceneObject</td>
</tr>
<tr>
<td>onEnterLiquid(string objId, string waterCoverage, string liquidType)</td>
<td>RigidShape</td>
</tr>
<tr>
<td>onLeaveLiquid(string objId, string liquidType)</td>
<td>RigidShape</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pauseThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAudio(int slot, SFXTrack track)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>playThread(int slot, string name=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>reset()</td>
<td>RigidShape</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string</td>
<td></td>
</tr>
<tr>
<td>Method Name</td>
<td>Class Name</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setAllMeshesHidden(bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCloaked(bool cloak)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setControl(bool controlled)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setDamageFlash(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageLevel(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageState(string state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageVector(Point3F vec)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDataBlock(GameBaseData data)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEnergyLevel(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool show)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAltTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAmmo(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageGenericTrigger(int slot, int trigger, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageLoaded(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageScriptAnimPrefix(int slot, string prefix)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTarget(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class Name</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInvincibleMode</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked</td>
<td>SimObject</td>
</tr>
<tr>
<td>setMeshHidden</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>SimObject</td>
</tr>
<tr>
<td>setRechargeRate</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setRepairRate</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setScale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways</td>
<td>NetObject</td>
</tr>
<tr>
<td>setShapeName</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setSkinName</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setSuperClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setThreadDir</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setThreadPosition</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setThreadTimeScale</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setTransform</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setVelocity</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setWhiteOut</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>skin</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>startFade</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>stopAudio</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>stopThread</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>unmount</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>unmountImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>validateCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
RigidShapeData Member List

This is the complete list of members for `RigidShapeData`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>bodyFriction</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>bodyRestitution</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>cameraDecay</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>cameraDefaultFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraLag</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>cameraMaxDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMaxFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraOffset</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>cameraRoll</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>category</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>checkDeployPos(TransformF txfm)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>collisionTol</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>computeCRC</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>contactTol</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td>Function/Property</td>
<td>Type</td>
</tr>
<tr>
<td>------------------</td>
<td>------</td>
</tr>
<tr>
<td>cubeReflectorDesc</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>Debris</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>debrisShapeName</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>density</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>destroyedLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>disabledLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>drag</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>dragForce</td>
<td>RigidShapeData</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dustEmitter</td>
<td>RigidShapeData</td>
</tr>
<tr>
<td>dustHeight</td>
<td>RigidShapeData</td>
</tr>
<tr>
<td>dustTrailEmitter</td>
<td>RigidShapeData</td>
</tr>
<tr>
<td>exitingWater</td>
<td>RigidShapeData</td>
</tr>
<tr>
<td>exitSplashSoundVelocity</td>
<td>RigidShapeData</td>
</tr>
<tr>
<td>Explosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>firstPersonOnly</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeployTransform(Point3F pos, Point3F normal)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td><code>getDynamicField</code>(int index)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getField</code>(int index)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType</code>(string fieldName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue</code>(string fieldName, int index=-1)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getGroup</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>hardImpactSound</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>hardImpactSpeed</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>hardSplashSoundVelocity</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>impactWaterEasy</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>impactWaterHard</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>impactWaterMedium</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>inheritEnergyFromMount</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>integration</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup</code>(SimGroup group)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField</code>(string fieldName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code>(string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInvincible</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>mass</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>massBox</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>massCenter</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>maxDamage</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>maxDrag</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>maxEnergy</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>mediumSplashSoundVelocity</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>minDrag</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>minImpactSpeed</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>minRollSpeed</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>observeThroughObject</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onAdd(GameBase obj)</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onCollision(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onDamage(ShapeBase obj, float delta)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onDestroyed(ShapeBase obj, string lastState)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onDisabled(ShapeBase obj, string lastState)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onEnabled(ShapeBase obj, string lastState)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onEndSequence(ShapeBase obj, int slot)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onForceUncloak(ShapeBase obj, string reason)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onImpact(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>onMount</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onNewDataBlock</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onRemove</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onTrigger</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onUnmount</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>reloadOnLocalClient</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>renderWhenDestroyed</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>repairRate</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCansave</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNameSpace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>shadowEnable</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>shadowMaxVisibleDistance</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>shadowProjectionDistance</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>shadowSize</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>shadowSphereAdjust</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>shapeFile</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>softImpactSound</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>softImpactSpeed</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>softSplashSoundVelocity</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>splashEmitter</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>splashFreqMod</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>splashVelEpsilon</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>triggerDustHeight</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>underwaterExplosion</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>useEyePoint</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>vertFactor</code></td>
<td><code>RigidShapeData</code></td>
</tr>
<tr>
<td><code>waterWakeSound</code></td>
<td><code>RigidShapeData</code></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
### River Member List

This is the complete list of members for River, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>baseColor</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clarity</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>cubemap</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>density</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>depthGradientMax</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>depthGradientTex</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>distortEndDist</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>distortFullDepth</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>distortStartDist</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>EditorOpen</code></td>
<td>River</td>
</tr>
<tr>
<td><code>emissive</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>FlowMagnitude</code></td>
<td>River</td>
</tr>
<tr>
<td><code>foamAmbientLerp</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>foamDir</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>foamMaxDepth</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>foamOpacity</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>foamRippleInfluence</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>foamSpeed</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>foamTex</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>foamTexScale</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>fresnelBias</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>fresnelPower</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>fullReflect</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getEulerRotation()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
</tbody>
</table>

*Note: The table above lists various functions and their corresponding classes. Each function belongs to a specific class indicated in the right column.*
<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>River [static]</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>River [static]</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>liquidType()</td>
<td>WaterObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>LowLODDistance</td>
<td>River</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>Node</td>
<td>River</td>
</tr>
<tr>
<td>overallFoamOpacity</td>
<td>WaterObject</td>
</tr>
<tr>
<td>overallRippleMagnitude</td>
<td>WaterObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td><code>overallWaveMagnitude</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>position</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>reflectDetailAdjust</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>reflectivity</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>reflectMaxRateMs</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>reflectNormalUp</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>reflectPriority</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>reflectTexSize</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>regenerate()</code></td>
<td><code>River</code></td>
</tr>
<tr>
<td><code>rippleDir</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>rippleMagnitude</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>rippleSpeed</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>rippleTex</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>rippleTexScale</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>rotation</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>scale</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>scopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>SegmentLength</code></td>
<td><code>River</code></td>
</tr>
<tr>
<td><code>setBatchSize(float meters)</code></td>
<td><code>River</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNameSpace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setMaxDivisionSize(float meters)</td>
<td>River</td>
</tr>
<tr>
<td>setMetersPerSegment(float meters)</td>
<td>River</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNodeDepth(int idx, float meters)</td>
<td>River</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>showNodes(TransformF txfm)</td>
<td>River [static]</td>
</tr>
<tr>
<td>showRiver</td>
<td>River [static]</td>
</tr>
<tr>
<td>showSpline</td>
<td>River [static]</td>
</tr>
<tr>
<td>showWalls</td>
<td>River [static]</td>
</tr>
<tr>
<td>showWireframe</td>
<td>River [static]</td>
</tr>
<tr>
<td>soundAmbience</td>
<td>WaterObject</td>
</tr>
<tr>
<td>specularColor</td>
<td>WaterObject</td>
</tr>
<tr>
<td>specularPower</td>
<td>WaterObject</td>
</tr>
<tr>
<td>SubdivideLength</td>
<td>River</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>---------------</td>
<td>------------</td>
</tr>
<tr>
<td>underwaterColor</td>
<td>WaterObject</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>useOcclusionQuery</td>
<td>WaterObject</td>
</tr>
<tr>
<td>viscosity</td>
<td>WaterObject</td>
</tr>
<tr>
<td>waterFogDensity</td>
<td>WaterObject</td>
</tr>
<tr>
<td>waterFogDensityOffset</td>
<td>WaterObject</td>
</tr>
<tr>
<td>waveDir</td>
<td>WaterObject</td>
</tr>
<tr>
<td>waveMagnitude</td>
<td>WaterObject</td>
</tr>
<tr>
<td>waveSpeed</td>
<td>WaterObject</td>
</tr>
<tr>
<td>wetDarkening</td>
<td>WaterObject</td>
</tr>
<tr>
<td>wetDepth</td>
<td>WaterObject</td>
</tr>
<tr>
<td>wireframe</td>
<td>WaterObject [static]</td>
</tr>
</tbody>
</table>
SaveFileDialog Member List

This is the complete list of members for `SaveFileDialog`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>changePath</td>
<td>FileDialog</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>defaultFile</td>
<td>FileDialog</td>
</tr>
<tr>
<td>defaultPath</td>
<td>FileDialog</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Execute()</td>
<td>FileDialog</td>
</tr>
<tr>
<td>fileName</td>
<td>FileDialog</td>
</tr>
<tr>
<td>filters</td>
<td>FileDialog</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getld()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>OverwritePrompt</td>
<td>SaveFileDialog</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>save(fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(fieldName, type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>title</td>
<td>FileDialog</td>
</tr>
</tbody>
</table>
ScatterSky Member List

This is the complete list of members for ScatterSky, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>ambientScale</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>applyChanges()</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>attenuationRatio</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>azimuth</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>brightness</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>castShadows</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>cookie</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>elevation</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>exposure</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>fadeStartDistance</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>flareScale</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>flareType</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>fogScale</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getScale()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>includeLightmappedGeometryInShadow</code></td>
<td><code>ScatterSky</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>isRenderable</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject</td>
<td>NetObject</td>
</tr>
<tr>
<td>lastSplitTerrainOnly</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>logWeight</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>mieScattering</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>moonAzimuth</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>moonElevation</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>moonEnabled</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>moonLightColor</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>moonMat</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>moonScale</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>nightColor</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>nightCubemap</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>nightFogColor</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>numSplits</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>overDarkFactor</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>Method/Property</td>
<td>Type</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rayleighScattering</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>representedInLightmap</td>
<td>ScatterSky</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>shadowDarkenColor</code></td>
<td><code>ScatterSky</code></td>
</tr>
<tr>
<td><code>shadowDistance</code></td>
<td><code>ScatterSky</code></td>
</tr>
<tr>
<td><code>shadowSoftness</code></td>
<td><code>ScatterSky</code></td>
</tr>
<tr>
<td><code>shadowType</code></td>
<td><code>ScatterSky</code></td>
</tr>
<tr>
<td><code>skyBrightness</code></td>
<td><code>ScatterSky</code></td>
</tr>
<tr>
<td><code>sunScale</code></td>
<td><code>ScatterSky</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>texSize</code></td>
<td><code>ScatterSky</code></td>
</tr>
<tr>
<td><code>unmount</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountObject</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>useNightCubemap</code></td>
<td><code>ScatterSky</code></td>
</tr>
</tbody>
</table>
# SceneObject Member List

This is the complete list of members for `SceneObject`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code> (SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>call</code> (string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient</code> (NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump</code> (bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCanSave</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClientObject</code> ()</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getDebugInfo</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField</code> (int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getEulerRotation</code> ()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function/Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool name=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject()</td>
<td>SceneObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# ScriptGroup Member List

This is the complete list of members for `ScriptGroup`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>acceptsAsChild(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>add(SimObject objects...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>bringToFront(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>callOnChildren(string method, string args...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>callOnChildrenNoRecurse(string method, string args...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clear()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>findObjectByInternalName(string internalName, bool searchChildren=false)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMember(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>listObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>onAdd(SimObjectId ID)</code></td>
<td><code>ScriptGroup</code></td>
</tr>
<tr>
<td><code>onObjectAdded(SimObject object)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>onObjectRemoved(SimObject object)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>onRemove(SimObjectId ID)</code></td>
<td><code>ScriptGroup</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>pushToBack(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>remove(SimObject objects...)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>reorderChild(SimObject child1, SimObject child2)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>setHidden (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>sort (string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
This is the complete list of members for **ScriptMsgListener**, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>onAdd()</code></td>
<td><code>ScriptMsgListener</code></td>
</tr>
<tr>
<td><code>onAddToQueue(string queue)</code></td>
<td><code>ScriptMsgListener</code></td>
</tr>
<tr>
<td><code>onMessageObjectReceived(string queue, Message msg)</code></td>
<td><code>ScriptMsgListener</code></td>
</tr>
<tr>
<td><code>onMessageReceived(string queue, string event, string data)</code></td>
<td><code>ScriptMsgListener</code></td>
</tr>
<tr>
<td><code>onRemove()</code></td>
<td><code>ScriptMsgListener</code></td>
</tr>
<tr>
<td><code>onRemoveFromQueue(string queue)</code></td>
<td><code>ScriptMsgListener</code></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>save</code></td>
<td>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
</tr>
<tr>
<td><code>schedule</code></td>
<td>(float time, string method, string args...)</td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td>(string name)</td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td>(string fieldName, string type)</td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td>(string fieldName, string value, int index=-1)</td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td>(string fileName)</td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td>(string newInternalName)</td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td>(bool state=true)</td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td>(bool state=true)</td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setName</code></td>
<td>(string newName)</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td>(string name)</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
This is the complete list of members for `ScriptObject`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code> (SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>call</code> (string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clone</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump</code> (bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField</code> (int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField</code> (int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType</code> (string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td><code>getFieldValue</code> (string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>onAdd(SimObjectId ID)</code></td>
<td>ScriptObject</td>
</tr>
<tr>
<td><code>onRemove(SimObjectId ID)</code></td>
<td>ScriptObject</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td>Set the class namespace</td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td>Set whether the object is only for editing</td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td>Set the field type</td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td>Set the field value</td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td>Set the file name</td>
</tr>
<tr>
<td><code> setHidden</code></td>
<td>Set whether the object is hidden</td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td>Set the internal name</td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td>Set whether the object is expanded</td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td>Set whether the object is selected</td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td>Set whether the object is locked</td>
</tr>
<tr>
<td><code>setName</code></td>
<td>Set the name</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td>Set whether name change is allowed</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td>Set the superclass namespace</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>Get the superclass</td>
</tr>
</tbody>
</table>
# SFXAmbience Member List

This is the complete list of members for SFXAmbience, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>call</td>
<td>SimObject, string method, string args...</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete</td>
<td>SimObject</td>
</tr>
<tr>
<td>dopplerFactor</td>
<td>SFXAmbience</td>
</tr>
<tr>
<td>dump</td>
<td>SimObject, bool detailed=false</td>
</tr>
<tr>
<td>dumpClassHierarchy</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods</td>
<td>SimObject</td>
</tr>
<tr>
<td>environment</td>
<td>SFXAmbience</td>
</tr>
<tr>
<td>getCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField</td>
<td>SimObject, int index</td>
</tr>
<tr>
<td>getDynamicFieldCount</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField</td>
<td>SimObject, int index</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>rolloffFactor</td>
<td>SFXAmbience</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>schedule</code> (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCanSave</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>soundTrack</code></td>
<td>SFXAmbience</td>
</tr>
<tr>
<td><code>states</code></td>
<td>SFXAmbience</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
</tbody>
</table>
# SFXController Member List

This is the complete list of members for SFXController, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addMarker(String name, float pos)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>addParameter(SFXParameter parameter)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>description</td>
<td>SFXSource</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>findObjectByInternalName</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getAttenuatedVolume()</code></td>
<td><code>SFXSource</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getCurrentSlot()</code></td>
<td><code>SFXController</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFadeInTime()</code></td>
<td><code>SFXSource</code></td>
</tr>
<tr>
<td><code>getFadeOutTime()</code></td>
<td><code>SFXSource</code></td>
</tr>
<tr>
<td><code>getField</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObject</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getObjectIndex</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getParameter</code></td>
<td><code>SFXSource</code></td>
</tr>
<tr>
<td><code>getParameter(int index)</code></td>
<td><code>SFXSource</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>getParameterCount()</code></td>
<td><code>SFXSource</code></td>
</tr>
<tr>
<td><code>getPitch()</code></td>
<td><code>SFXSource</code></td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getStatus()</code></td>
<td><code>SFXSource</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getVolume()</code></td>
<td><code>SFXSource</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMember(SimObject obj)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isPaused()</code></td>
<td><code>SFXSource</code></td>
</tr>
<tr>
<td><code>isPlaying()</code></td>
<td><code>SFXSource</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isStopped()</code></td>
<td><code>SFXSource</code></td>
</tr>
<tr>
<td><code>listObjects()</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>onObjectAdded(SimObject object)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>onObjectRemoved(SimObject object)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>onParameterValueChange(SFXParameter parameter)</code></td>
<td><code>SFXSource</code></td>
</tr>
<tr>
<td>Method Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>onStatusChange</code></td>
<td>(SFXStatus newStatus)</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>pause</code></td>
<td>(float fadeOutTime=-1.f)</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>pushToBack</code></td>
<td>(SimObject obj)</td>
</tr>
<tr>
<td><code>remove</code></td>
<td>(SimObject objects...)</td>
</tr>
<tr>
<td><code>removeParameter</code></td>
<td>(SFXParameter parameter)</td>
</tr>
<tr>
<td><code>reorderChild</code></td>
<td>(SimObject child1, SimObject child2)</td>
</tr>
<tr>
<td><code>save</code></td>
<td>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
</tr>
<tr>
<td><code>schedule</code></td>
<td>(float time, string method, string args...)</td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td>(string name)</td>
</tr>
<tr>
<td><code>setCone</code></td>
<td>(float innerAngle, float outerAngle, float outsideVolume)</td>
</tr>
<tr>
<td><code>setCurrentSlot</code></td>
<td>(int index)</td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setFadeTimes</code></td>
<td>(float fadeInTime, float fadeOutTime)</td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td>(string fieldName, string type)</td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td>(string fieldName, string value, int index=-1)</td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td>(string fileName)</td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td>(string newInternalName)</td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td>(bool state=true)</td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td>(bool state=true)</td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setName</code></td>
<td>(string newName)</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td>(bool value=true)</td>
</tr>
</tbody>
</table>

SFXSource:

- `onStatusChange`:
- `parentGroup`:
- `pause`:
- `persistentId`:
- `pushToBack`:
- `remove`:
- `removeParameter`:
- `reorderChild`:
- `save`:
- `schedule`:
- `setCanSave`:
- `setClassNamespace`:
- `setCone`:
- `setCurrentSlot`:
- `setEditorOnly`:
- `setFadeTimes`:
- `setFieldType`:
- `setFieldValue`:
- `setFilename`:
- `setHidden`:
- `setInternalName`:
- `setIsExpanded`:
- `setIsSelected`:
- `setLocked`:
- `setName`:
- `setNameChangeAllowed`:
<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>setPitch</code> (float pitch)</td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setTransform</code> (Point3F position, Point3F direction)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>void setTransform(Point3F position)</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>void play(float fadeInTime)</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>setVolume</code> (float volume)</td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>sort</code> (string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>statusCallback</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>stop</code> (float fadeOutTime=-1.f)</td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>trace</code></td>
<td>SFXController</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
This is the complete list of members for `SFXDescription`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>coneInsideAngle</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>coneOutsideAngle</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>coneOutsideVolume</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>fadeInEase</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>fadeInTime</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>fadeLoops</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>fadeOutEase</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>fadeOutTime</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>is3D</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isLooping</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isStreaming</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>maxDistance</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>parameters</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>pitch</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>priority</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>referenceDistance</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>reloadOnLocalClient()</code></td>
<td><code>SimDataBlock</code></td>
</tr>
<tr>
<td><code>REVERB_DIRECTHFAUTO</code></td>
<td><code>SFXDescription</code> [static]</td>
</tr>
<tr>
<td><code>REVERB_INSTANCE0</code></td>
<td><code>SFXDescription</code> [static]</td>
</tr>
<tr>
<td><code>REVERB_INSTANCE1</code></td>
<td><code>SFXDescription</code> [static]</td>
</tr>
<tr>
<td><code>REVERB_INSTANCE2</code></td>
<td><code>SFXDescription</code> [static]</td>
</tr>
<tr>
<td><code>REVERB_INSTANCE3</code></td>
<td><code>SFXDescription</code> [static]</td>
</tr>
<tr>
<td><code>REVERB_ROOMAUTO</code></td>
<td><code>SFXDescription</code> [static]</td>
</tr>
<tr>
<td><code>REVERB_ROOMHFAUTO</code></td>
<td><code>SFXDescription</code> [static]</td>
</tr>
<tr>
<td><code>reverbAirAbsorptionFactor</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>reverbDirect</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>reverbDirectHF</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>reverbDopplerFactor</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>reverbExclusion</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>reverbExclusionLFRatio</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>reverbFlags</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td><code>reverbObstruction</code></td>
<td><code>SFXDescription</code></td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>reverbObstructionLFRatio</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>reverbOcclusion</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>reverbOcclusionDirectRatio</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>reverbOcclusionLFRatio</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>reverbOcclusionRoomRatio</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>reverbOutsideVolumeHF</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>reverbReverbRolloffFactor</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>reverbRoom</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>reverbRoomHF</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>reverbRoomRolloffFactor</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>rolloffFactor</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString='')</td>
<td>SimObject</td>
</tr>
<tr>
<td>scatterDistance</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>sourceGroup</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>streamPacketSize</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>streamReadAhead</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>useCustomReverb</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>useHardware</td>
<td>SFXDescription</td>
</tr>
<tr>
<td>volume</td>
<td>SFXDescription</td>
</tr>
</tbody>
</table>
# SFXEmitter Member List

This is the complete list of members for **SFXEmitter**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>coneInsideAngle</code></td>
<td>SFXEmitter</td>
</tr>
<tr>
<td><code>coneOutsideAngle</code></td>
<td>SFXEmitter</td>
</tr>
<tr>
<td><code>coneOutsideVolume</code></td>
<td>SFXEmitter</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>fadeInTime</code></td>
<td>SFXEmitter</td>
</tr>
<tr>
<td><code>fadeOutTime</code></td>
<td>SFXEmitter</td>
</tr>
<tr>
<td><code>fileName</code></td>
<td>SFXEmitter</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getEulerRotation()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getScale()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSource()</td>
<td>SFXEmitter</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>is3D</td>
<td>SFXEmitter</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isLooping</td>
<td>SFXEmitter</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>SFXEmitter</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>SFXEmitter</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function/Member</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isStreaming</code></td>
<td><code>SFXEmitter</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>maxDistance</code></td>
<td><code>SFXEmitter</code></td>
</tr>
<tr>
<td><code>mountNode</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountPID</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountPos</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountRot</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>pitch</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>play()</code></td>
<td><code>SFXEmitter</code></td>
</tr>
<tr>
<td><code>playOnAdd</code></td>
<td><code>SFXEmitter</code></td>
</tr>
<tr>
<td><code>position</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>referenceDistance</code></td>
<td><code>SFXEmitter</code></td>
</tr>
<tr>
<td><code>renderColorInnerCone</code></td>
<td><code>SFXEmitter</code></td>
</tr>
<tr>
<td></td>
<td><code>[static]</code></td>
</tr>
<tr>
<td><code>renderColorOuterCone</code></td>
<td><code>SFXEmitter</code></td>
</tr>
<tr>
<td></td>
<td><code>[static]</code></td>
</tr>
<tr>
<td><code>renderColorOutsideVolume</code></td>
<td><code>SFXEmitter</code></td>
</tr>
<tr>
<td></td>
<td><code>[static]</code></td>
</tr>
<tr>
<td><code>renderColorPlayingInRange</code></td>
<td><code>SFXEmitter</code></td>
</tr>
<tr>
<td></td>
<td><code>[static]</code></td>
</tr>
<tr>
<td><code>renderColorPlayingOutOfRange</code></td>
<td><code>SFXEmitter</code></td>
</tr>
<tr>
<td></td>
<td><code>[static]</code></td>
</tr>
<tr>
<td><code>renderColorRangeSphere</code></td>
<td><code>SFXEmitter</code></td>
</tr>
<tr>
<td></td>
<td><code>[static]</code></td>
</tr>
<tr>
<td><code>renderColorStoppedInRange</code></td>
<td><code>SFXEmitter</code></td>
</tr>
<tr>
<td></td>
<td><code>[static]</code></td>
</tr>
<tr>
<td><code>renderColorStoppedOutOfRange</code></td>
<td><code>SFXEmitter</code></td>
</tr>
<tr>
<td></td>
<td><code>[static]</code></td>
</tr>
<tr>
<td><code>renderEmitters</code></td>
<td><code>SFXEmitter</code></td>
</tr>
<tr>
<td></td>
<td><code>[static]</code></td>
</tr>
<tr>
<td><code>renderPointDistance</code></td>
<td><code>SFXEmitter</code></td>
</tr>
<tr>
<td></td>
<td><code>[static]</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>renderRadialIncrements</td>
<td>SFXEmitter</td>
</tr>
<tr>
<td>renderSweepIncrements</td>
<td>SFXEmitter</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>scatterDistance</td>
<td>SFXEmitter</td>
</tr>
<tr>
<td>schedule (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient (NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassName (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType (string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScale (Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways ()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassName (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTransform (TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method/Property</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>sourceGroup</td>
<td>SFXEmitter</td>
</tr>
<tr>
<td>stop()</td>
<td>SFXEmitter</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>track</td>
<td>SFXEmitter</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>useTrackDescriptionOnly</td>
<td>SFXEmitter</td>
</tr>
<tr>
<td>volume</td>
<td>SFXEmitter</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
### SFXEnvironment Member List

This is the complete list of members for `SFXEnvironment`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>airAbsorptionHF</td>
<td><code>SFXEnvironment</code></td>
</tr>
<tr>
<td>assignFieldsFrom()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>call()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>canSave</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>class</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>className</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>clone()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>decayHFRatio</td>
<td><code>SFXEnvironment</code></td>
</tr>
<tr>
<td>decayLFRatio</td>
<td><code>SFXEnvironment</code></td>
</tr>
<tr>
<td>decayTime</td>
<td><code>SFXEnvironment</code></td>
</tr>
<tr>
<td>deepClone()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>delete()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>density</td>
<td><code>SFXEnvironment</code></td>
</tr>
<tr>
<td>diffusion</td>
<td><code>SFXEnvironment</code></td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>dumpMethods()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>echoDepth</td>
<td><code>SFXEnvironment</code></td>
</tr>
<tr>
<td>echoTime</td>
<td><code>SFXEnvironment</code></td>
</tr>
<tr>
<td>envDiffusion</td>
<td><code>SFXEnvironment</code></td>
</tr>
<tr>
<td>Member</td>
<td>Return Type</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>envSize</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>flags</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>HFReference</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function/Variable</td>
<td>Type</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>isMemberOfClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>modulationDepth</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>modulationTime</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>reflections</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>reflectionsDelay</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>reflectionsPan</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>reloadOnLocalClient</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>reverb</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>REVERB_CORE0</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>REVERB_CORE1</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>REVERB_DECAYHFLIMIT</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>REVERB_DECAYTIMESCALE</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>REVERB_ECHOTIMESCALE</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>REVERB_HIGHQUALITYDPL2REVERB</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>REVERB_HIGHQUALITYREVERB</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>REVERB_MODULATIONTIMESCALE</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>REVERB_REFLECTIONSDELYSCALE</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>REVERB_REFLECTIONSSCALE</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>REVERB_REVERBDELYSCALE</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>REVERB_REVERBSCALE</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>reverbDelay</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>reverbPan</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>room</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>roomHF</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>roomLF</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td>roomRolloffFactor</td>
<td>SFXEnvironment</td>
</tr>
<tr>
<td><strong>save</strong> (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>schedule</strong> (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setCanSave</strong> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setClassNamespace</strong> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setEditorOnly</strong> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setFieldType</strong> (string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setFieldValue</strong> (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setFilename</strong> (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>** setHidden** (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setInternalName</strong> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setIsExpanded</strong> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setIsSelected</strong> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setLocked</strong> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setName</strong> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setNameChangeAllowed</strong> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setSuperClassNamespace</strong> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>superClass</strong></td>
<td>SimObject</td>
</tr>
</tbody>
</table>
# SFXFMODEvent Member List

This is the complete list of members for SFXFMODEvent, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>description</td>
<td>SFXTrack</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>fmodGroup</td>
<td>SFXFMODEvent</td>
</tr>
<tr>
<td>fmodName</td>
<td>SFXFMODEvent</td>
</tr>
<tr>
<td>fmodParameterRanges</td>
<td>SFXFMODEvent</td>
</tr>
<tr>
<td>fmodParameterValues</td>
<td>SFXFMODEvent</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parameters</td>
<td>SFXTrack</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCansave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## SFXFMODEventGroup Member List

This is the complete list of members for SFXFMODEventGroup, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>fmodGroup</td>
<td>SFXFMODEventGroup</td>
</tr>
<tr>
<td>fmodName</td>
<td>SFXFMODEventGroup</td>
</tr>
<tr>
<td>fmodProject</td>
<td>SFXFMODEventGroup</td>
</tr>
<tr>
<td>freeData()</td>
<td>SFXFMODEventGroup</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isDataLoaded()</td>
<td>SFXFMODEventGroup</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>loadData(bool loadStreams=true, bool loadSamples=true)</td>
<td>SFXFMODEventGroup</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>
**SFXFMODEventSource Member List**

This is the complete list of members for `SFXFMODEventSource`, including all inherited members.

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>addMarker(String name, float pos)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>addParameter(SFXParameter parameter)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimSet</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>description</td>
<td>SFXSource</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>findObjectByInternalName</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getAttenuatedVolume()</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFadeInTime()</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>getFadeOutTime()</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getParameter(int index)</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>getParameterCount()</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getPitch()</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>getRandom()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getStatus()</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getVolume()</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMember(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isPaused()</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>isPlaying()</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isStopped()</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>listObjects()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>onObjectAdded(SimObject object)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onObjectRemoved(SimObject object)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>onParameterValueChange(SFXParameter parameter)</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>onStatusChange(SFXStatus newStatus)</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td>Method</td>
<td>Object</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>pause(float fadeOutTime=-1.f)</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>pushToBack(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>remove(SimObject objects...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>removeParameter(SFXParameter parameter)</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>reorderChild(SimObject child1, SimObject child2)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCone(float innerAngle, float outerAngle, float outsideVolume)</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFadeTimes(float fadeInTime, float fadeOutTime)</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code> setHidden(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setPitch(float pitch)</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>setTransform(Point3F position, Point3F direction) void</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>setTransform(Point3F position)</code> void</td>
<td></td>
</tr>
<tr>
<td><code>play(float fadeInTime)</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>setVolume(float volume)</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>sort(string callbackFunction)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>statusCallback</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>stop(float fadeOutTime=-1.f)</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## SFXFMODProject Member List

This is the complete list of members for **SFXFMODProject**, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom (SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call (string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>fileName</td>
<td>SFXFMODProject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td><code>getFieldType</code> (string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue</code> (string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup</code> (SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField</code> (string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass</code> (string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod</code> (string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>mediaPath</code></td>
<td>SFXFMODProject</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>reloadOnLocalClient</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>save</code> (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule</code> (float time, string method, string)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Synopsis</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>args...</td>
<td></td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## SFXParameter Member List

This is the complete list of members for *SFXParameter*, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>channel</code></td>
<td><code>SFXParameter</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>defaultValue</code></td>
<td><code>SFXParameter</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>description</code></td>
<td><code>SFXParameter</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getParameterName()</td>
<td>SFXParameter</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onUpdate()</td>
<td>SFXParameter</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>range</td>
<td>SFXParameter</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>reset()</td>
<td></td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>value</td>
<td>SFXParameter</td>
</tr>
</tbody>
</table>
SFXPlayList Member List

This is the complete list of members for **SFXPlayList**, including all inherited members.

<table>
<thead>
<tr>
<th>Function/Property</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delayTimeIn</td>
<td>SFXPlayList</td>
</tr>
<tr>
<td>delayTimeInVariance</td>
<td>SFXPlayList</td>
</tr>
<tr>
<td>delayTimeOut</td>
<td>SFXPlayList</td>
</tr>
<tr>
<td>delayTimeOutVariance</td>
<td>SFXPlayList</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>description</td>
<td>SFXTrack</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>fadeTimeIn</td>
<td>SFXPlayList</td>
</tr>
<tr>
<td>fadeTimeInVariance</td>
<td>SFXPlayList</td>
</tr>
<tr>
<td>fadeTimeOut</td>
<td>SFXPlayList</td>
</tr>
<tr>
<td>fadeTimeOutVariance</td>
<td>SFXPlayList</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>------------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>loopMode</code></td>
<td><code>SFXPlayList</code></td>
</tr>
<tr>
<td><code>maxDistance</code></td>
<td><code>SFXPlayList</code></td>
</tr>
<tr>
<td><code>maxDistanceVariance</code></td>
<td><code>SFXPlayList</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>numSlotsToPlay</code></td>
<td><code>SFXPlayList</code></td>
</tr>
<tr>
<td><code>parameters</code></td>
<td><code>SFXTrack</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>pitchScale</code></td>
<td><code>SFXPlayList</code></td>
</tr>
<tr>
<td><code>pitchScaleVariance</code></td>
<td><code>SFXPlayList</code></td>
</tr>
<tr>
<td><code>random</code></td>
<td><code>SFXPlayList</code></td>
</tr>
<tr>
<td><code>referenceDistance</code></td>
<td><code>SFXPlayList</code></td>
</tr>
<tr>
<td><code>referenceDistanceVariance</code></td>
<td><code>SFXPlayList</code></td>
</tr>
<tr>
<td><code>reloadOnLocalClient()</code></td>
<td><code>SimDataBlock</code></td>
</tr>
<tr>
<td><code>repeatCount</code></td>
<td><code>SFXPlayList</code></td>
</tr>
<tr>
<td><code>replay</code></td>
<td><code>SFXPlayList</code></td>
</tr>
<tr>
<td><code>save</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>save</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><code>reloadOnLocalClient()</code></td>
<td></td>
</tr>
<tr>
<td><code>repeatCount</code></td>
<td></td>
</tr>
<tr>
<td><code>replay</code></td>
<td></td>
</tr>
<tr>
<td><code>save</code></td>
<td></td>
</tr>
<tr>
<td><code>schedule</code></td>
<td></td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td></td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td></td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td></td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td></td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td></td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td></td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td></td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>state</code></td>
<td>SFXPlayList</td>
</tr>
<tr>
<td><code>stateMode</code></td>
<td>SFXPlayList</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>trace</code></td>
<td>SFXPlayList</td>
</tr>
<tr>
<td><code>track</code></td>
<td>SFXPlayList</td>
</tr>
<tr>
<td><code>transitionIn</code></td>
<td>SFXPlayList</td>
</tr>
<tr>
<td><code>transitionOut</code></td>
<td>SFXPlayList</td>
</tr>
<tr>
<td><code>volumeScale</code></td>
<td>SFXPlayList</td>
</tr>
<tr>
<td><code>volumeScaleVariance</code></td>
<td>SFXPlayList</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
SFXProfile Member List

This is the complete list of members for SFXProfile, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>description</td>
<td>SFXTrack</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>fileName</td>
<td>SFXProfile</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSoundDuration()</td>
<td>SFXProfile</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parameters</td>
<td>SFXTrack</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>preload</td>
<td>SFXProfile</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td><code>save</code></td>
<td>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
</tr>
<tr>
<td><code>schedule</code></td>
<td>(float time, string method, string args...)</td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td>(string name)</td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td>(string fieldName, string type)</td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td>(string fieldName, string value, int index=-1)</td>
</tr>
<tr>
<td><code>setName</code></td>
<td>(string fileName)</td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td>(string fileName)</td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td>(string newName)</td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td>(bool state=true)</td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td>(bool state=true)</td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setName</code></td>
<td>(string newName)</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td>(string name)</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td></td>
</tr>
</tbody>
</table>
# SFXSound Member List

This is the complete list of members for **SFXSound**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>acceptsAsChild(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>add(SimObject objects...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>addMarker(String name, float pos)</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>addParameter(SFXParameter parameter)</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>bringToFront(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>callOnChildren(string method, string args...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>callOnChildrenNoRecurse(string method, string args...)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clear()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deleteAllObjects()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>description</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>dumpMethods</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>findObjectByInternalName</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getAttenuatedVolume</code></td>
<td><code>SFXSource</code></td>
</tr>
<tr>
<td><code>getCanSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCount</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getDebugInfo</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDuration</code></td>
<td><code>SFXSound</code></td>
</tr>
<tr>
<td><code>getDynamicField</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFadeInTime</code></td>
<td><code>SFXSource</code></td>
</tr>
<tr>
<td><code>getFadeOutTime</code></td>
<td><code>SFXSource</code></td>
</tr>
<tr>
<td><code>getField</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFullCount</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObject</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getObjectIndex</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>getParameter</code></td>
<td><code>SFXSource</code></td>
</tr>
<tr>
<td><code>getParameter</code></td>
<td><code>SFXSource</code></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>getParameterCount()</td>
<td>SFXSource</td>
</tr>
<tr>
<td>getPitch()</td>
<td>SFXSource</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SFXSound</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getStatus()</td>
<td>SFXSource</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getVolume()</td>
<td>SFXSource</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isPaused()</td>
<td>SFXSource</td>
</tr>
<tr>
<td>isPlaying()</td>
<td>SFXSource</td>
</tr>
<tr>
<td>isReady()</td>
<td>SFXSound</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isStopped()</td>
<td>SFXSource</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>onParameterValueChange(SFXParameter parameter)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>onStatusChange(SFXStatus newStatus)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pause(float fadeOutTime=-1.f)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimObject</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>removeParameter(SFXParameter parameter)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCone(float innerAngle, float outerAngle, float outsideVolume)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFadeTimes(float fadeInTime, float fadeOutTime)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPitch(float pitch)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>setPosition(float position)</td>
<td>SFXSound</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTransform(Point3F position, Point3F direction)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setVolume(float volume)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>statusCallback</td>
<td>SFXSource</td>
</tr>
<tr>
<td>stop(float fadeOutTime=-1.f)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# SFXSource Member List

This is the complete list of members for **SFXSource**, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>acceptsAsChild</strong> (SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td><strong>add</strong> (SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td><strong>addMarker</strong> (String name, float pos)</td>
<td>SFXSource</td>
</tr>
<tr>
<td><strong>addParameter</strong> (SFXParameter parameter)</td>
<td>SFXSource</td>
</tr>
<tr>
<td><strong>assignFieldsFrom</strong> (SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>assignPersistentId</strong> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>bringToFront</strong> (SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td><strong>call</strong> (string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>callOnChildren</strong> (string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td><strong>callOnChildrenNoRecurse</strong> (string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td><strong>canSave</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>canSaveDynamicFields</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>class</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>className</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>clear</strong> ()</td>
<td>SimSet</td>
</tr>
<tr>
<td><strong>clone</strong> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>deepClone</strong> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>delete</strong> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>deleteAllObjects</strong> ()</td>
<td>SimSet</td>
</tr>
<tr>
<td><strong>description</strong></td>
<td>SFXSource</td>
</tr>
<tr>
<td><strong>dump</strong> (bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>dumpClassHierarchy</strong> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>dumpGroupHierarchy</strong> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>findObjectByName(string internalName, bool searchChildren=false)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getAttenuatedVolume()</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFadeInTime()</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>getFadeOutTime()</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFullCount()</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getObjectIndex(SimObject obj)</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getParameter(int index)</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td><code>getParameterCount()</code></td>
<td>SFXSource</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>getPitch()</td>
<td>SFXSource</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getStatus()</td>
<td>SFXSource</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getVolume()</td>
<td>SFXSource</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isPaused()</td>
<td>SFXSource</td>
</tr>
<tr>
<td>isPlaying()</td>
<td>SFXSource</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isStopped()</td>
<td>SFXSource</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onParameterValueChanged(SFXParameter parameter)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>onStatusChange(SFXStatus newStatus)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pause(float fadeOutTime=-1.f)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>removeParameter(SFXParameter parameter)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCone(float innerAngle, float outerAngle, float outsideVolume)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFadeTimes(float fadeInTime, float fadeOutTime)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPitch(float pitch)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>setTransform(Point3F position, Point3F direction) void setTransform(Point3F position) void play(float fadeInTime)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>setVolume(float volume)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>sort(string callbackFunction)</td>
<td>SimSet</td>
</tr>
<tr>
<td>statusCallback</td>
<td>SFXSource</td>
</tr>
<tr>
<td>stop(float fadeOutTime=-1.f)</td>
<td>SFXSource</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>
**SFXSpace Member List**

This is the complete list of members for *SFXSpace*, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>edge</td>
<td>SFXSpace</td>
</tr>
<tr>
<td>getCANSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>SFXSpace [static]</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>SFXSpace [static]</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>plane</td>
<td>SFXSpace</td>
</tr>
<tr>
<td>point</td>
<td>SFXSpace</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTransform(TransformF)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>soundAmbience</td>
<td>SFXSpace</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject(SceneObject)</td>
<td>SceneObject</td>
</tr>
</tbody>
</table>
This is the complete list of members for **SFXState**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>activate()</code></td>
<td><strong>SFXState</strong></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>canSave()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>class</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>className</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>deactivate()</code></td>
<td><strong>SFXState</strong></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>disable()</code></td>
<td><strong>SFXState</strong></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>enable()</code></td>
<td><strong>SFXState</strong></td>
</tr>
<tr>
<td><code>excludedStates</code></td>
<td><strong>SFXState</strong></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>includedStates</td>
<td>SFXState</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isActive()</td>
<td>SFXState</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isDisabled()</td>
<td>SFXState</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>onActivate()</td>
<td>SFXState</td>
</tr>
<tr>
<td>onDeactivate()</td>
<td>SFXState</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
## SFXTrack Member List

This is the complete list of members for SFXTrack, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>description</td>
<td>SFXTrack</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>getFieldType</code> (string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue</code> (string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getld</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup</code> (SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField</code> (string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass</code> (string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod</code> (string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>parameters</code></td>
<td>SFXTrack</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentld</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>reloadOnLocalClient</code></td>
<td>SimDataBlock</td>
</tr>
<tr>
<td><code>save</code> (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule</code> (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
### ShaderData Member List

This is the complete list of members for ShaderData, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>defines</code></td>
<td>ShaderData</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>DXPixelShaderFile</code></td>
<td>ShaderData</td>
</tr>
<tr>
<td><code>DXVertexShaderFile</code></td>
<td>ShaderData</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>locked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>OGLPixelShaderFile</code></td>
<td>ShaderData</td>
</tr>
<tr>
<td><code>OGLVertexShaderFile</code></td>
<td>ShaderData</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>pixVersion</code></td>
<td>ShaderData</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>reload()</td>
<td>ShaderData</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>useDevicePixVersion</td>
<td>ShaderData</td>
</tr>
</tbody>
</table>
# ShapeBase Member List

This is the complete list of members for `ShapeBase`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>applyDamage(float amount)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>applyImpulse(Point3F pos, Point3F vec)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>GameBase::applyImpulse(Point3F pos, VectorF vel)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>applyRadialImpulse(Point3F origin, float radius, float magnitude)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>applyRepair(float amount)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>boundingBox</code></td>
<td><code>GameBase</code> [static]</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canCloak()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>changeMaterial(string mapTo, Material oldMat, Material newMat)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dataBlock</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>destroyThread(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMeshVisibility()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getAIRepairPoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClass()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getControllingClient()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getControllingObject()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageFlash()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamagePercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageState()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDefaultCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEnergyLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEnergyPercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getEyePoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeTransform()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeVector()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getImageAltTrigger(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageAmmo(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageGenericTrigger(int slot, int trigger)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageLoaded(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageScriptAnimPrefix(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageSkinTag(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageState(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageTarget(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageTrigger(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getLookAtPoint(float distance=2000, int typeMask=0xFFFFFFFF)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMaxDamage()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getModelFile()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountSlot(ShapeBaseImageData)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzlePoint(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzleVector(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPendingImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRechargeRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRepairRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getShapeName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSkinName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSlotTransform(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTargetCount()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTargetName(int index)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getVelocity()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getWhiteOut()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hasImageState(int slot, string state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isAIControlled</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isCloaked()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDestroyed()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDisabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEnabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isHidden()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageFiring(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageMounted(ShapeBaseImageData image)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>ShapeBase</code> [static]</td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>ShapeBase</code> [static]</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountImage(ShapeBaseImageData image, int slot, bool loaded=true, string skinTag=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SceneObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SceneObject</td>
</tr>
<tr>
<td>pauseThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAudio(int slot, SFXTrack track)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>playThread(int slot, string name=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setAllMeshesHidden(bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCloaked(bool cloak)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setControl(bool controlled)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setDamageFlash(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageLevel(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageState(string state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageVector(Point3F vec)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDataBlock(GameBaseData data)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEnergyLevel(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool show)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAltTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAmmo(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageGenericTrigger(int slot, int trigger, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageLoaded(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageScriptAnimPrefix(int slot, string prefix)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTarget(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInvincibleMode(float time, float speed)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Category</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>setMeshHidden</code> (string name, bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setRechargeRate</code> (float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setRepairRate</code> (float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setScale</code> (Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setScopeAlways</code> ()</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setShapeName</code> (string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setSkinName</code> (string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setThreadDir</code> (int slot, bool fwd)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setThreadPosition</code> (int slot, float pos)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setThreadTimeScale</code> (int slot, float scale)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setTransform</code> (TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setVelocity</code> (Point3F vel)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setWhiteOut</code> (float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>skin</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>startFade</code> (int time, int delay, bool fadeOut)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>stopAudio</code> (int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>stopThread</code> (int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>unmount</code> ()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>unmountImage</code> (int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>unmountObject</code> (SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>validateCameraFov</code> (float fov)</td>
<td>ShapeBase</td>
</tr>
</tbody>
</table>
# ShapeBaseData Member List

This is the complete list of members for `ShapeBaseData`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>cameraDefaultFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMaxDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMaxFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>category</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>checkDeployPos(TransformF tfm)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>computeCRC</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cubeReflectorDesc</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>Debris</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>debrisShapeName</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>density</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>destroyedLevel</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>disabledLevel</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>drag</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Explosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>firstPersonOnly</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeployTransform(Point3F pos, Point3F normal)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>inheritEnergyFromMount</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInvincible</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>locked()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>mass</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>maxDamage</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>maxEnergy</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>observeThroughObject</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onAdd(GameBase obj)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onCollision(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onDamage(ShapeBase obj, float delta)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onDestroyed(ShapeBase obj, string lastState)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onDisabled(ShapeBase obj, string lastState)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onEnabled(ShapeBase obj, string lastState)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onEndSequence(ShapeBase obj, int slot)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onForceUncloak(ShapeBase obj, string reason)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onImpact(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>onMount</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>GameBase obj, SceneObject</td>
<td></td>
</tr>
<tr>
<td>mountObj, int node</td>
<td></td>
</tr>
<tr>
<td>onNewDataBlock</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>GameBase obj</td>
<td></td>
</tr>
<tr>
<td>onRemove</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>GameBase obj</td>
<td></td>
</tr>
<tr>
<td>onTrigger</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>ShapeBase obj, int index, bool</td>
<td></td>
</tr>
<tr>
<td>state</td>
<td></td>
</tr>
<tr>
<td>onUnmount</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>GameBase obj, SceneObject</td>
<td></td>
</tr>
<tr>
<td>mountObj, int node</td>
<td></td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>reloadOnLocalClient</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>renderWhenDestroyed</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>repairRate</td>
<td></td>
</tr>
<tr>
<td>save(string fileName, bool</td>
<td>SimObject</td>
</tr>
<tr>
<td>selectedOnly=false, string</td>
<td></td>
</tr>
<tr>
<td>preAppendString=&quot;&quot;</td>
<td></td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>float time, string method, string</td>
<td></td>
</tr>
<tr>
<td>args...</td>
<td></td>
</tr>
<tr>
<td>setCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>bool value=true</td>
<td></td>
</tr>
<tr>
<td>setClassName</td>
<td>SimObject</td>
</tr>
<tr>
<td>string name</td>
<td></td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>bool value=true</td>
<td></td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>string fieldName, string type</td>
<td></td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>string fieldName, string value,</td>
<td></td>
</tr>
<tr>
<td>int index=-1</td>
<td></td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>string fileName</td>
<td></td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>bool value=true</td>
<td></td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>string newInternalName</td>
<td></td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>bool state=true</td>
<td></td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>bool state=true</td>
<td></td>
</tr>
<tr>
<td>setLocked</td>
<td>SimObject</td>
</tr>
<tr>
<td>bool value=true</td>
<td></td>
</tr>
<tr>
<td>setName</td>
<td>SimObject</td>
</tr>
<tr>
<td>string newName</td>
<td></td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>SimObject</td>
</tr>
<tr>
<td>bool value=true</td>
<td></td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>shadowEnable</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowMaxVisibleDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowProjectionDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSize</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSphereAdjust</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shapeFile</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>underwaterExplosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>useEyePoint</td>
<td>ShapeBaseData</td>
</tr>
</tbody>
</table>
ShapeBaseImageData Member List

This is the complete list of members for *ShapeBaseImageData*, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>accuFire</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>animateAllShapes</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>animateOnServer</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>camShakeAmp</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>camShakeFreq</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>casing</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>category</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>cloakable</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>computeCRC</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>correctMuzzleVector</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>correctMuzzleVectorTP</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>emap</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>eyeOffset</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>eyeRotation</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>firstPerson</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>imageAnimPrefix</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>imageAnimPrefixFP</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class/Interface</td>
</tr>
<tr>
<td>------------------------------------------------------------</td>
<td>------------------------------------------------------</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>lightBrightness</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>lightColor</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>lightDuration</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>lightRadius</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>lightType</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mass</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>maxConcurrentSounds</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>minEnergy</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>mountPoint</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>offset</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>onAdd(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onMount(ShapeBase obj, int slot, float dt)</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>GameBaseData::onMount(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onNewDataBlock(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onRemove(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onUnmount(ShapeBase obj, int slot, float dt)</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>Function/Method</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>GameBaseData::onUnmount</strong>(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td><strong>parentGroup</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>persistentId</strong></td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>Projectile</strong></td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td><strong>reloadOnLocalClient()</strong></td>
<td>SimDataBlock</td>
</tr>
<tr>
<td><strong>rotation</strong></td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td><strong>save</strong>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>schedule</strong>(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>scriptAnimTransitionTime</strong></td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td><strong>setCanSave</strong>(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setClassNamespace</strong>(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setEditorOnly</strong>(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setFieldType</strong>(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setFieldValue</strong>(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setFilename</strong>(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setHidden</strong>(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setInternalName</strong>(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setIsExpanded</strong>(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setIsSelected</strong>(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setLocked</strong>(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setName</strong>(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setNameChangeAllowed</strong>(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setSuperClassNamespace</strong>(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>shakeCamera</strong></td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>Variable</td>
<td>Data Type</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>shapeFile</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>shapeFileFP</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>shellExitDir</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>shellExitVariance</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>shellVelocity</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateAllowImageChange</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateAlternateFire</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateDirection</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateEjectShell</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateEmitter</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateEmitterNode</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateEmitterTime</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateEnergyDrain</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateFire</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateIgnoreLoadedForReady</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateLoadedFlag</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateName</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateRecoil</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateReload</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateScaleAnimation</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateScaleAnimationFP</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateScaleShapeSequence</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateScript</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateSequence</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateSequenceNeverTransition</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateSequenceRandomFlash</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateSequenceTransitionIn</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateSequenceTransitionOut</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateSequenceTransitionTime</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>stateShapeSequence</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateSound</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateSpinThread</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTimeoutValue</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionGeneric0In</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionGeneric0Out</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionGeneric1In</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionGeneric1Out</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionGeneric2In</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionGeneric2Out</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionGeneric3In</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionGeneric3Out</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionOnAltTriggerDown</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionOnAltTriggerUp</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionOnAmmo</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionOnLoaded</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionOnMotion</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionOnNoAmmo</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionOnNoMotion</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionOnNoTarget</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionOnNotLoaded</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionOnNotWet</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionOnTarget</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionOnTimeout</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionOnTriggerDown</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateTransitionOnTriggerUp</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>Property</td>
<td>Value</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>stateTransitionOnWet</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>stateWaitForTimeout</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>useEyeNode</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>useRemainderDT</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>usesEnergy</td>
<td>ShapeBaseImageData</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# SimDataBlock Member List

This is the complete list of members for SimDataBlock, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>reloadOnLocalClient()</code></td>
<td><code>SimDataBlock</code></td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNameSpace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
This is the complete list of members for **SimGroup**, including all inherited members.

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>acceptsAsChild</code> (SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>add</code> (SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>assignFieldsFrom</code> (SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>bringToFront</code> (SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>call</code> (string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>callOnChildren</code> (string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>callOnChildrenNoRecurse</code> (string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clear</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>clone</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deleteAllObjects</code></td>
<td>SimSet</td>
</tr>
<tr>
<td><code>dump</code> (bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>findObjectByInternalName</code> (string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td><code>getCanSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>isMember(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>listObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onObjectAdded(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>onObjectRemoved(SimObject object)</td>
<td>SimSet</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pushToBack(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>remove(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>reorderChild(SimObject child1, SimObject child2)</td>
<td>SimSet</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNameSpace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td><code>isSelected</code></td>
<td>bool state=true</td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td>bool value=true</td>
</tr>
<tr>
<td><code>setName</code></td>
<td>string newName</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td>bool value=true</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td>string name</td>
</tr>
<tr>
<td><code>sort</code></td>
<td>string callbackFunction</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td></td>
</tr>
</tbody>
</table>
# SimObject Member List

This is the complete list of members for SimObject, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getFieldValue</code> (string <code>fieldName</code>, int <code>index=-1</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup</code> (SimGroup <code>group</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField</code> (string <code>fieldName</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code> (string <code>name</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass</code> (string <code>className</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod</code> (string <code>methodName</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>save</code> (string <code>fileName</code>, bool <code>selectedOnly=false</code>, string <code>preAppendString=&quot;&quot;</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule</code> (float <code>time</code>, string <code>method</code>, string <code>args...</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCanSave</code> (bool <code>value=true</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string <code>name</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool <code>value=true</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>setFieldType</code> (string <code>fieldName</code>, string <code>type</code>)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue</code> (string <code>fieldName</code>, string <code>value</code>, int <code>index=-1</code>)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code> (string <code>fileName</code>)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden</code> (bool <code>value=true</code>)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code> (string <code>newInternalName</code>)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool <code>state=true</code>)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool <code>state=true</code>)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code> (bool <code>value=true</code>)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code> (string <code>newName</code>)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool <code>value=true</code>)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string <code>name</code>)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
SimpleMessageEvent Member List

This is the complete list of members for SimpleMessageEvent, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg(NetConnection con, string message)</td>
<td>SimpleMessageEvent [static]</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
SimpleNetObject Member List

This is the complete list of members for SimpleNetObject, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>call</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>scopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNameSpace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setMessage(string msg)</code></td>
<td><code>SimpleNetObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNameSpace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
</tbody>
</table>
SimSet Member List

This is the complete list of members for **SimSet**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>acceptsAsChild(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>add(SimObject objects...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>bringToFront(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>callOnChildren(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>callOnChildrenNoRecurse(string method, string args...)</td>
<td>SimSet</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear()</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deleteAllObjects()</td>
<td>SimSet</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>findObjectByInternalName(string internalName, bool searchChildren=false)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFullCount()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObject(int index)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getObjectIndex(SimObject obj)</td>
<td>SimSet</td>
</tr>
<tr>
<td>getRandom()</td>
<td>SimSet</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Argument/Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><code>isMember</code></td>
<td><code>SimObject obj</code></td>
</tr>
<tr>
<td><code>isMemberOfClass</code></td>
<td><code>string className</code></td>
</tr>
<tr>
<td><code>isMethod</code></td>
<td><code>string methodName</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code></td>
<td></td>
</tr>
<tr>
<td><code>isSelected</code></td>
<td></td>
</tr>
<tr>
<td><code>listObjects</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>onObjectAdded</code></td>
<td><code>SimObject object</code></td>
</tr>
<tr>
<td><code>onObjectRemoved</code></td>
<td><code>SimObject object</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>pushToBack</code></td>
<td><code>SimObject obj</code></td>
</tr>
<tr>
<td><code>remove</code></td>
<td><code>SimObject objects...</code></td>
</tr>
<tr>
<td><code>reorderChild</code></td>
<td><code>SimObject child1, SimObject child2</code></td>
</tr>
<tr>
<td><code>save</code></td>
<td><code>SimObject fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;</code></td>
</tr>
<tr>
<td><code>schedule</code></td>
<td><code>float time, string method, string args...</code></td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td><code>bool value=true</code></td>
</tr>
<tr>
<td><code>setClassNameSpace</code></td>
<td><code>string name</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td><code>bool value=true</code></td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td><code>string fieldName, string type</code></td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td><code>string fieldName, string value, int index=-1</code></td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td><code>string fileName</code></td>
</tr>
<tr>
<td><code> setHidden</code></td>
<td><code>bool value=true</code></td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td><code>string newInternalName</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td><code>bool state=true</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>sort(string callbackFunction)</code></td>
<td><code>SimSet</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
</tbody>
</table>
**SimXMLDocument Member List**

This is the complete list of members for `SimXMLDocument`, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>addComment</code> (string comment)</td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>addData</code> (string text)</td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>addHeader</code> ()</td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>addNewElement</code> (string name)</td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>addText</code> (string text)</td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom</code> (SimObject fromObject)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>attribute</code> (string attributeName)</td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>attributeExists</code> (string attributeName)</td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>attributeF32</code> (string attributeName)</td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>attributeS32</code> (string attributeName)</td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>call</code> (string method, string args...)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clear</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>clearError</code> ()</td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>clone</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump</code> (bool detailed=false)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>elementValue()</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>firstAttribute()</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getData()</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getErrorDesc()</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getText()</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>isField</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>lastAttribute</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>loadFile</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>nextAttribute</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>nextSiblingElement</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>parse</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>popElement</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>prevAttribute</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>pushChildElement</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>pushFirstChildElement</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>pushNewElement</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>readComment</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>removeText</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>reset</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>save</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>saveFile</code></td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>schedule</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>setAttribute</code> (string attributeName, string value)</td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>setCanSave</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldNamespace</code> (string fieldName, string type)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldvalue</code> (string fieldName, string value, int index=-1)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setObjectAttributes</code> (string objectID)</td>
<td><code>SimXMLDocument</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
</tbody>
</table>
# SkyBox Member List

This is the complete list of members for **SkyBox**, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>assignFieldsFrom</strong> (SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>call</strong>(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>clearScopeToClient</strong>(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>drawBottom</td>
<td>SkyBox</td>
</tr>
<tr>
<td><strong>dump</strong>(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>fogBandHeight</td>
<td>SkyBox</td>
</tr>
<tr>
<td><strong>getCanSave</strong>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>getClassName</strong>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>getClassNamespace</strong>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>getClientObject</strong>()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField**(int index)**</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectName()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>SkyBox</code> [static]</td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>SkyBox</code> [static]</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>Material</code></td>
<td><code>SkyBox</code></td>
</tr>
<tr>
<td><code>mountNode</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountPID</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>postApply</td>
<td>SkyBox</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td><code>SceneObject</code></td>
</tr>
</tbody>
</table>
## SpawnSphere Member List

This is the complete list of members for **SpawnSphere**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>applyDamage(float amount)</code></td>
<td><strong>ShapeBase</strong></td>
</tr>
<tr>
<td><code>applyImpulse(Point3F pos, Point3F vec)</code></td>
<td><strong>ShapeBase</strong></td>
</tr>
<tr>
<td><code>GameBase::applyImpulse(Point3F pos, VectorF vel)</code></td>
<td><strong>GameBase</strong></td>
</tr>
<tr>
<td><code>applyRadialImpulse(Point3F origin, float radius, float magnitude)</code></td>
<td><strong>GameBase</strong></td>
</tr>
<tr>
<td><code>applyRepair(float amount)</code></td>
<td><strong>ShapeBase</strong></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>autoSpawn</code></td>
<td><strong>SpawnSphere</strong></td>
</tr>
<tr>
<td><code>boundingBox</code></td>
<td><strong>GameBase</strong></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>canCloak()</code></td>
<td><strong>ShapeBase</strong></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>changeMaterial(string mapTo, Material oldMat, Material newMat)</code></td>
<td><strong>ShapeBase</strong></td>
</tr>
<tr>
<td><code>class</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>className</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td><strong>NetObject</strong></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>dataBlock</code></td>
<td><strong>GameBase</strong></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>dataBlock</code></td>
<td><strong>GameBase</strong></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>destroyThread(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMeshVisibility()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getAIRepairPoint()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getCameraFov()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getControllingClient()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getControllingObject()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamageFlash()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamageLevel()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamagePercent()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamageState()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDataBlock()</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDefaultCameraFov()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getEnergyLevel()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getEnergyPercent()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getEulerRotation()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getEyePoint()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td><code>getEyeTransform()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getEyeVector()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getImageAltTrigger(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageAmmo(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageGenericTrigger(int slot, int trigger)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageLoaded(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageScriptAnimPrefix(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageSkinTag(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageState(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageTarget(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageTrigger(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getLookAtPoint(float distance=2000, int typeMask=0xFFFFFFFF)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMaxDamage()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getModelFile()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Function Name</td>
<td>Type</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>getMountedImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountSlot(ShapeBaseImageData image)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzlePoint(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzleVector(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getobjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPendingImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getPosition(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRechargeRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRepairRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getShapeName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSkinName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSlotTransform(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTargetCount()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTargetName(int index)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td><code>getVelocity()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getWhiteOut()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hasImageState(int slot, string state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>indoorWeight</code></td>
<td><code>SpawnSphere</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isAIControlled</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isCloaked()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDestroyed()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDisabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEnabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isHidden()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageFiring(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageMounted(ShapeBaseImageData image)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>isRenderable</td>
<td>SpawnSphere [static]</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>SpawnSphere [static]</td>
</tr>
<tr>
<td>isSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountImage(ShapeBaseImageData image, int slot, bool loaded=true, string skinTag=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAdd(int objectId)</td>
<td>SpawnSphere</td>
</tr>
<tr>
<td>outdoorWeight</td>
<td>SpawnSphere</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pauseThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAudio(int slot, SFXTrack track)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>playThread(int slot, string name=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>radius</td>
<td>SpawnSphere</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td><code>scale</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>schedule(time, method, args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>scopeToClient(client)</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setAllMeshesHidden(hide)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setCameraFov(fov)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setCanSave(value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace(name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCloaked(cloak)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setControl(controlled)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>setDamageFlash(level)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setDamageLevel(level)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setDamageState(state)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setDamageVector(vec)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setDataBlock(data)</code></td>
<td>GameBaseData</td>
</tr>
<tr>
<td><code>setEditorOnly(value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEnergyLevel(level)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setFieldType(fieldName, type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue(fieldName, value, index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename(fileName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden(show)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setImageAltTrigger(slot, state)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setImageAmmo(slot, state)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setImageGenericTrigger(slot, trigger, state)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setImageLoaded(slot, state)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setImageScriptAnimPrefix(slot, prefix)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>setImageTarget(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInvincibleMode(float time, float speed)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setMeshHidden(string name, bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setRechargeRate(float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setRepairRate(float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setShapeName(string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setSkinName(string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setThreadDir(int slot, bool fwd)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setThreadPosition(int slot, float pos)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setThreadTimeScale(int slot, float scale)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setVelocity(Point3F vel)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setWhiteOut(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>skin</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>spawnClass</td>
<td>SpawnSphere</td>
</tr>
<tr>
<td>spawnDatablock</td>
<td>SpawnSphere</td>
</tr>
<tr>
<td>Method/Property</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>spawnObject(string additionalProps)</td>
<td>SpawnSphere</td>
</tr>
<tr>
<td>spawnProperties</td>
<td>SpawnSphere</td>
</tr>
<tr>
<td>spawnScript</td>
<td>SpawnSphere</td>
</tr>
<tr>
<td>spawnTransform</td>
<td>SpawnSphere</td>
</tr>
<tr>
<td>sphereWeight</td>
<td>SpawnSphere</td>
</tr>
<tr>
<td>startFade(int time, int delay, bool fadeOut)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>stopAudio(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>stopThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>validateCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# Splash Member List

This is the complete list of members for Splash, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>applyImpulse(Point3F pos, VectorF vel)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>applyRadialImpulse(Point3F origin, float radius, float magnitude)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>boundingBox</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dataBlock</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>Splash [static]</code></td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>Splash [static]</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>mountNode</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>mountObject</code> (SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountPID</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountPos</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountRot</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>name</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>position</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>rotation</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>save</code> (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>scale</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>schedule</code> (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>scopeToClient</code> (NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setCanSave</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setControl</code> (bool controlled)</td>
<td>GameBase</td>
</tr>
<tr>
<td><code>setDataBlock</code> (GameBaseData data)</td>
<td>GameBase</td>
</tr>
<tr>
<td><code>setEditorOnly</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType</code> (string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue</code> (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code> (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
</tbody>
</table>
# SplashData Member List

This is the complete list of members for SplashData, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>acceleration</td>
<td>SplashData</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>category</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>colors</td>
<td>SplashData</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delayMS</td>
<td>SplashData</td>
</tr>
<tr>
<td>delayVariance</td>
<td>SplashData</td>
</tr>
<tr>
<td>delete()</td>
<td>SplashData</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>ejectionAngle</td>
<td>SplashData</td>
</tr>
<tr>
<td>ejectionFreq</td>
<td>SplashData</td>
</tr>
<tr>
<td>emitter</td>
<td>SplashData</td>
</tr>
<tr>
<td>Explosion</td>
<td>SplashData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>height</code></td>
<td>SplashData</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method/Field</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>lifetimeMS</td>
<td>SplashData</td>
</tr>
<tr>
<td>lifetimeVariance</td>
<td>SplashData</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>numSegments</td>
<td>SplashData</td>
</tr>
<tr>
<td>onAdd(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onMount(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onNewDataBlock(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onRemove(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onUnmount(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>ringLifetime</td>
<td>SplashData</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SplashData</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>soundProfile</code></td>
<td>SplashData</td>
</tr>
<tr>
<td><code>startRadius</code></td>
<td>SplashData</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>texFactor</code></td>
<td>SplashData</td>
</tr>
<tr>
<td><code>texture</code></td>
<td>SplashData</td>
</tr>
<tr>
<td><code>texWrap</code></td>
<td>SplashData</td>
</tr>
<tr>
<td><code>times</code></td>
<td>SplashData</td>
</tr>
<tr>
<td><code>velocity</code></td>
<td>SplashData</td>
</tr>
<tr>
<td><code>width</code></td>
<td>SplashData</td>
</tr>
</tbody>
</table>
# SpotLight Member List

This is the complete list of members for SpotLight, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>animate</td>
<td>LightBase</td>
</tr>
<tr>
<td>animationPeriod</td>
<td>LightBase</td>
</tr>
<tr>
<td>animationPhase</td>
<td>LightBase</td>
</tr>
<tr>
<td>animationType</td>
<td>LightBase</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>attenuationRatio</td>
<td>LightBase</td>
</tr>
<tr>
<td>brightness</td>
<td>LightBase</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>castShadows</td>
<td>LightBase</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>color</td>
<td>LightBase</td>
</tr>
<tr>
<td>cookie</td>
<td>LightBase</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>fadeStartDistance</td>
<td>LightBase</td>
</tr>
<tr>
<td>flareScale</td>
<td>LightBase</td>
</tr>
<tr>
<td>flareType</td>
<td>LightBase</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>includeLightmappedGeometryInShadow</td>
<td>LightBase</td>
</tr>
<tr>
<td>innerAngle</td>
<td>SpotLight</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEnabled</td>
<td>LightBase</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>SpotLight</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>SpotLight</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>lastSplitTerrainOnly</td>
<td>LightBase</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>logWeight</td>
<td>LightBase</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot,</td>
<td>SceneObject</td>
</tr>
<tr>
<td>TransformF txfm=MatrixF::Identity)</td>
<td></td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>numSplits</td>
<td>LightBase</td>
</tr>
<tr>
<td>outerAngle</td>
<td>SpotLight</td>
</tr>
<tr>
<td>overDarkFactor</td>
<td>LightBase</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pauseAnimation</td>
<td>LightBase</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAnimation(LightAnimData anim)</td>
<td>LightBase</td>
</tr>
<tr>
<td>playAnimation()</td>
<td>LightBase</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>priority</td>
<td>LightBase</td>
</tr>
<tr>
<td>range</td>
<td>SpotLight</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>representedInLightmap</td>
<td>LightBase</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>scopeToClient(NetConnection client)</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code> setHidden(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLightEnabled(bool state)</code></td>
<td>LightBase</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>shadowDarkenColor</code></td>
<td>LightBase</td>
</tr>
<tr>
<td>Method/Property</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>shadowDistance</td>
<td>LightBase</td>
</tr>
<tr>
<td>shadowSoftness</td>
<td>LightBase</td>
</tr>
<tr>
<td>shadowType</td>
<td>LightBase</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>texSize</td>
<td>LightBase</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
</tbody>
</table>
## StaticShape Member List

This is the complete list of members for `StaticShape`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>applyDamage(float amount)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>applyImpulse(Point3F pos, Point3F vec)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>GameBase::applyImpulse(Point3F pos, VectorF vel)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>applyRadialImpulse(Point3F origin, float radius, float magnitude)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>applyRepair(float amount)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>boundingBox</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canCloak()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>changeMaterial(string mapTo, Material oldMat, Material newMat)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dataBlock</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>destroyThread(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method Name</td>
<td>Class Name</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>dumpClassHierarchy</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMeshVisibility</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getAIRepairPoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getControllingClient()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getControllingObject()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageFlash()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamagePercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageState()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDefaultValue(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEnergyLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEnergyPercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEulerAngles()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getEyePoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeTransform()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeVector()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>getField( int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getImageAltTrigger( int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageAmmo(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageGenericTrigger(int slot, int trigger)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageLoaded(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageScriptAnimPrefix(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageSkinTag(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageState(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageTarget(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageTrigger(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getLookAtPoint(float distance=2000, int typeMask=0xFFFFFFFF)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMaxDamage()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getModelFile()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountSlot(ShapeBaseImageData)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzlePoint(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzleVector(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPendingImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRechargeRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRepairRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getShapeName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSkinName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSlotTransform(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTargetCount()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTargetName(int index)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getVelocity()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getWhiteOut()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hasImageState(int slot, string state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isAIControlled</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isCloaked()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDestroyed()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDisabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEnabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isHidden()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageFiring(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageMounted(ShapeBaseImageData image)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>StaticShape [static]</code></td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>StaticShape [static]</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountImage(ShapeBaseImageData image, int slot, bool loaded=true, string skinTag=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SceneObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pauseThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAudio(int slot, SFXTrack track)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>playThread(int slot, string name=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setAllMeshesHidden(bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCloaked</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setControl</td>
<td>GameBase</td>
</tr>
<tr>
<td>setDamageFlash</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageLevel</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageState</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageVector</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDataBlock</td>
<td>GameBase</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEnergyLevel</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAltTrigger</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAmmo</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageGenericTrigger</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageLoaded</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageScriptAnimPrefix</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTarget</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTrigger</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInvincibleMode</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>setMeshHidden</code> (string name, bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setRechargeRate</code> (float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setRepairRate</code> (float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setScale</code> (Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setScopeAlways</code> ()</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setShapeName</code> (string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setSkinName</code> (string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setThreadDir</code> (int slot, bool fwd)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setThreadPosition</code> (int slot, float pos)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setThreadTimeScale</code> (int slot, float scale)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setTransform</code> (TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setVelocity</code> (Point3F vel)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setWhiteOut</code> (float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>skin</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>startFade</code> (int time, int delay, bool fadeOut)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>stopAudio</code> (int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>stopThread</code> (int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>unmount</code> ()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>unmountImage</code> (int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>unmountObject</code> (SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>validateCameraFov</code> (float fov)</td>
<td>ShapeBase</td>
</tr>
</tbody>
</table>
# StaticShapeData Member List

This is the complete list of members for StaticShapeData, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>cameraDefaultFov</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cameraMaxDist</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cameraMaxFov</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cameraMinDist</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cameraMinFov</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>category</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>checkDeployPos(TransformF txfm)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>computeCRC</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cubeReflectorDesc</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>Debris</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>debrisShapeName</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>density</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>destroyedLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>disabledLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>Function Name</td>
<td>Class Name</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>drag</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dynamicType</code></td>
<td>StaticShapeData</td>
</tr>
<tr>
<td>Explosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>firstPersonOnly</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeployTransform(Point3F pos, Point3F normal)</code></td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Function/Method Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>inheritEnergyFromMount</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInvincible</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mass</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>maxDamage</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>maxEnergy</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>noIndividualDamage</td>
<td>StaticShapeData</td>
</tr>
<tr>
<td>observeThroughObject</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onAdd(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onCollision(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onDamage(ShapeBase obj, float delta)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onDestroyed(ShapeBase obj, string lastState)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onDisabled(ShapeBase obj, string lastState)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onEnabled(ShapeBase obj, string lastState)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onEndSequence(ShapeBase obj, int slot)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onForceUncloak(ShapeBase obj, string reason)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td><code>onImpact</code></td>
<td>ShapeBase object, SceneObject collider object, VectorF vector, float length</td>
</tr>
<tr>
<td><code>onMount</code></td>
<td>GameBase object, SceneObject mount object, int node</td>
</tr>
<tr>
<td><code>onNewDataBlock</code></td>
<td>GameBase object</td>
</tr>
<tr>
<td><code>onRemove</code></td>
<td>GameBase object</td>
</tr>
<tr>
<td><code>onTrigger</code></td>
<td>ShapeBase object, int index, bool state</td>
</tr>
<tr>
<td><code>onUnmount</code></td>
<td>GameBase object, SceneObject mount object, int node</td>
</tr>
<tr>
<td>parentGroup</td>
<td></td>
</tr>
<tr>
<td>persistentId</td>
<td></td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td></td>
</tr>
<tr>
<td>renderWhenDestroyed</td>
<td></td>
</tr>
<tr>
<td>repairRate</td>
<td></td>
</tr>
<tr>
<td><code>save</code></td>
<td>string fileName, bool selectedOnly=false, string prependString=&quot;&quot;</td>
</tr>
<tr>
<td><code>schedule</code></td>
<td>float time, string method, string args...</td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td>bool value=true</td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td>string name</td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td>bool value=true</td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td>string fieldName, string type</td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td>string fieldName, string value, int index=-1</td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td>string fileName</td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td>bool value=true</td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td>string newInternalName</td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td>bool state=true</td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td>bool state=true</td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td>bool value=true</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><strong>setName</strong> (string newName)</td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><strong>setNameChangeAllowed</strong> (bool value=true)</td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><strong>setSuperClassNamespace</strong> (string name)</td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><strong>shadowEnable</strong></td>
<td><strong>ShapeBaseData</strong></td>
</tr>
<tr>
<td><strong>shadowMaxVisibleDistance</strong></td>
<td><strong>ShapeBaseData</strong></td>
</tr>
<tr>
<td><strong>shadowProjectionDistance</strong></td>
<td><strong>ShapeBaseData</strong></td>
</tr>
<tr>
<td><strong>shadowSize</strong></td>
<td><strong>ShapeBaseData</strong></td>
</tr>
<tr>
<td><strong>shadowSphereAdjust</strong></td>
<td><strong>ShapeBaseData</strong></td>
</tr>
<tr>
<td><strong>shapeFile</strong></td>
<td><strong>ShapeBaseData</strong></td>
</tr>
<tr>
<td><strong>superClass</strong></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><strong>underwaterExplosion</strong></td>
<td><strong>ShapeBaseData</strong></td>
</tr>
<tr>
<td><strong>useEyePoint</strong></td>
<td><strong>ShapeBaseData</strong></td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
StreamObject Member List

This is the complete list of members for StreamObject, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>copyFrom(SimObject other)</td>
<td>StreamObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>getFieldType</code> (string fieldName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue</code> (string fieldName, int index=-1)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getPosition</code></td>
<td><code>StreamObject</code></td>
</tr>
<tr>
<td><code>getStatus</code></td>
<td><code>StreamObject</code></td>
</tr>
<tr>
<td><code>getStreamSize</code></td>
<td><code>StreamObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup</code> (SimGroup group)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEOF</code></td>
<td><code>StreamObject</code></td>
</tr>
<tr>
<td><code>isEOS</code></td>
<td><code>StreamObject</code></td>
</tr>
<tr>
<td><code>isExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField</code> (string fieldName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass</code> (string className)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod</code> (string methodName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>readLine()</code></td>
<td>StreamObject</td>
</tr>
<tr>
<td><code>readLongString(int maxLength)</code></td>
<td>StreamObject</td>
</tr>
<tr>
<td><code>readString()</code></td>
<td>StreamObject</td>
</tr>
<tr>
<td><code>readSTString(bool caseSensitive=false)</code></td>
<td>StreamObject</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setPosition(int newPosition)</code></td>
<td>StreamObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>writeLine(string line)</code></td>
<td>StreamObject</td>
</tr>
<tr>
<td><code>writeLongString(int maxLength, string string)</code></td>
<td>StreamObject</td>
</tr>
<tr>
<td><code>writeString(string string, int maxLength=256)</code></td>
<td>StreamObject</td>
</tr>
</tbody>
</table>
# Sun Member List

This is the complete list of members for **Sun**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ambient</td>
<td>Sun</td>
</tr>
<tr>
<td>animate</td>
<td>Sun</td>
</tr>
<tr>
<td>apply</td>
<td>Sun</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>attenuationRatio</td>
<td>Sun</td>
</tr>
<tr>
<td>azimuth</td>
<td>Sun</td>
</tr>
<tr>
<td>brightness</td>
<td>Sun</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>castShadows</td>
<td>Sun</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>color</td>
<td>Sun</td>
</tr>
<tr>
<td>cookie</td>
<td>Sun</td>
</tr>
<tr>
<td>coronaEnabled</td>
<td>Sun</td>
</tr>
<tr>
<td>coronaMaterial</td>
<td>Sun</td>
</tr>
<tr>
<td>coronaScale</td>
<td>Sun</td>
</tr>
<tr>
<td>coronaTint</td>
<td>Sun</td>
</tr>
<tr>
<td>coronaUseLightColor</td>
<td>Sun</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>elevation</td>
<td>Sun</td>
</tr>
<tr>
<td>fadeStartDistance</td>
<td>Sun</td>
</tr>
<tr>
<td>flareScale</td>
<td>Sun</td>
</tr>
<tr>
<td>flareType</td>
<td>Sun</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getScale()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>includeLightmappedGeometryInShadow</code></td>
<td><code>Sun</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass</code> (string className)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod</code> (string methodName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted</code> ()</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>Sun</code></td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>Sun</code></td>
</tr>
<tr>
<td><code>isSelected</code> ()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject</code> ()</td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>lastSplitTerrainOnly</code></td>
<td><code>Sun</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>logWeight</code></td>
<td><code>Sun</code></td>
</tr>
<tr>
<td><code>mountNode</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountObject</code> (SceneObject objB, int slot,</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>TransformF txfm=MatrixF::Identity)</td>
<td></td>
</tr>
<tr>
<td><code>mountPID</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountPos</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountRot</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>numSplits</code></td>
<td><code>Sun</code></td>
</tr>
<tr>
<td><code>overDarkFactor</code></td>
<td><code>Sun</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>position</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>representedInLightmap</code></td>
<td><code>Sun</code></td>
</tr>
<tr>
<td><code>rotation</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><strong>save</strong> (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>scale</strong></td>
<td>SceneObject</td>
</tr>
<tr>
<td><strong>schedule</strong> (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>scopeToClient</strong> (NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td><strong>setCanSave</strong> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setClassNamespace</strong> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setEditorOnly</strong> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setFieldType</strong> (string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setFieldValue</strong> (string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setFilename</strong> (string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setHidden</strong> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setInternalName</strong> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setIsExpanded</strong> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setIsSelected</strong> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setLocked</strong> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setName</strong> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setNameChangeAllowed</strong> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setScale</strong> (Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><strong>setScopeAlways</strong> ()</td>
<td>NetObject</td>
</tr>
<tr>
<td><strong>setSuperClassNamespace</strong> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setTransform</strong> (TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><strong>shadowDarkenColor</strong></td>
<td>Sun</td>
</tr>
<tr>
<td><strong>shadowDistance</strong></td>
<td>Sun</td>
</tr>
<tr>
<td><strong>shadowSoftness</strong></td>
<td>Sun</td>
</tr>
<tr>
<td>shadowType</td>
<td>Sun</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>texSize</td>
<td>Sun</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
</tbody>
</table>
# TCPObjec Member List

This is the complete list of members for `TCPObjec`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Call</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>connect(string address)</code></td>
<td>TCPObject</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>disconnect()</code></td>
<td>TCPObject</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>listen(int port)</td>
<td>TCPObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onConnected()</td>
<td>TCPObject</td>
</tr>
<tr>
<td>onConnectFailed()</td>
<td>TCPObject</td>
</tr>
<tr>
<td>onConnectionRequest(string address, string ID)</td>
<td>TCPObject</td>
</tr>
<tr>
<td>onDisconnect()</td>
<td>TCPObject</td>
</tr>
<tr>
<td>onDNSFailed()</td>
<td>TCPObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>onDNSResolved()</td>
<td>TCPObject</td>
</tr>
<tr>
<td>onLine(string line)</td>
<td>TCPObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentld</td>
<td>SimObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>send(string data)</td>
<td>TCPObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>
**TerrainBlock Member List**

This is the complete list of members for **TerrainBlock**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>baseTexSize</code></td>
<td><code>TerrainBlock</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>castShadows</code></td>
<td><code>TerrainBlock</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>createNew</code></td>
<td><code>TerrainBlock</code></td>
</tr>
<tr>
<td><code>debugRender</code></td>
<td><code>TerrainBlock</code></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong><code>deepClone()</code></strong></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><code>exportHeightMap(string filename,[string format])</code></td>
<td><code>TerrainBlock</code></td>
</tr>
<tr>
<td><code>exportLayerMaps(string filePrefix,[string format])</code></td>
<td><code>TerrainBlock</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getEulerRotation()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getScale()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>import(String terrainName, String heightMap, F32 metersPerPixel, F32 heightScale, String materials, String opacityLayers)</code></td>
<td><code>TerrainBlock</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>TerrainBlock</code></td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>TerrainBlock</code></td>
</tr>
<tr>
<td>Function/Membership</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>lightMapSize</td>
<td>TerrainBlock</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName)</td>
<td>TerrainBlock</td>
</tr>
<tr>
<td>SceneObject::save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>screenError</td>
<td>TerrainBlock</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setScale</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setScopeAlways</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setTransform</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>squareSize</code></td>
<td>TerrainBlock</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>terrainFile</code></td>
<td>TerrainBlock</td>
</tr>
<tr>
<td><code>unmount</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>unmountObject</code></td>
<td>SceneObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# TerrainMaterial Member List

This is the complete list of members for TerrainMaterial, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>detailDistance</td>
<td>TerrainMaterial</td>
</tr>
<tr>
<td>detailMap</td>
<td>TerrainMaterial</td>
</tr>
<tr>
<td>detailSize</td>
<td>TerrainMaterial</td>
</tr>
<tr>
<td>detailStrength</td>
<td>TerrainMaterial</td>
</tr>
<tr>
<td>diffuseMap</td>
<td>TerrainMaterial</td>
</tr>
<tr>
<td>diffuseSize</td>
<td>TerrainMaterial</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getld()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>normalMap</td>
<td>TerrainMaterial</td>
</tr>
<tr>
<td>parallaxScale</td>
<td>TerrainMaterial</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>useSideProjection</td>
<td>TerrainMaterial</td>
</tr>
</tbody>
</table>
# TimeOfDay Member List

This is the complete list of members for `TimeOfDay`, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>addTimeOfDayEvent(float elevation, string identifier)</code></td>
<td><code>TimeOfDay</code></td>
</tr>
<tr>
<td><code>animate(float elevation, float degreesPerSecond)</code></td>
<td><code>TimeOfDay</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>axisTilt</code></td>
<td><code>TimeOfDay</code></td>
</tr>
<tr>
<td><code>azimuthOverride</code></td>
<td><code>TimeOfDay</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dayLength</code></td>
<td><code>TimeOfDay</code></td>
</tr>
<tr>
<td><code>dayScale</code></td>
<td><code>TimeOfDay</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getEulerRotation()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getScale()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup</code>(SimGroup <code>group</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField</code>(string <code>fieldName</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code>(string <code>name</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass</code>(string <code>className</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod</code>(string <code>methodName</code>)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>TimeOfDay</code></td>
</tr>
<tr>
<td><code>isRenderEnabled()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>TimeOfDay</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>nightScale</td>
<td>TimeOfDay</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>play</td>
<td>TimeOfDay</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setDayLength(float seconds)</code></td>
<td>TimeOfDay</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setPlay(bool enabled)</td>
<td>TimeOfDay</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTimeOfDay(float time)</td>
<td>TimeOfDay</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>startTime</td>
<td>TimeOfDay</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>time</td>
<td>TimeOfDay</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
### Trigger Member List

This is the complete list of members for *Trigger*, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>applyImpulse(Point3F pos, VectorF vel)</code></td>
<td><strong>GameBase</strong></td>
</tr>
<tr>
<td><code>applyRadialImpulse(Point3F origin, float radius, float magnitude)</code></td>
<td><strong>GameBase</strong></td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>boundingBox</code></td>
<td><strong>GameBase</strong> [static]</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>class</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>className</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td><strong>NetObject</strong></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>dataBlock</code></td>
<td><strong>GameBase</strong></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>enterCommand</code></td>
<td><strong>Trigger</strong></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td><strong>SimObject</strong></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getDataBlock()</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getEulerRotation()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getName()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getNumObjects()</code></td>
<td>Trigger</td>
</tr>
<tr>
<td><code>getObject(int index)</code></td>
<td>Trigger</td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>Trigger [static]</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>Trigger [static]</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>leaveCommand</td>
<td>Trigger</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(Sc, int slot, TF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SceneObject</td>
</tr>
<tr>
<td>onAdd(int objectId)</td>
<td>Trigger</td>
</tr>
<tr>
<td>onRemove(int objectId)</td>
<td>Trigger</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>polyhedron</td>
<td>Trigger</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>renderTriggers</td>
<td>Trigger</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setControl(bool controlled)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setDataBlock(GameBaseData data)</td>
<td>GameBase</td>
</tr>
<tr>
<td>Method Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setFieldType</td>
<td>(string fieldName, string type)</td>
</tr>
<tr>
<td>setValue</td>
<td>(string fieldName, string value, int index=-1)</td>
</tr>
<tr>
<td>setFilename</td>
<td>(string fileName)</td>
</tr>
<tr>
<td>setHidden</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setInternalName</td>
<td>(string newInternalName)</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>(bool state=true)</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>(bool state=true)</td>
</tr>
<tr>
<td>setLocked</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setName</td>
<td>(string newName)</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>setScale</td>
<td>(Point3F scale)</td>
</tr>
<tr>
<td>setScopeAlways</td>
<td>()</td>
</tr>
<tr>
<td>setSuperClassNamespace</td>
<td>(string name)</td>
</tr>
<tr>
<td>setTransform</td>
<td>(TransformF txfm)</td>
</tr>
<tr>
<td>superClass</td>
<td></td>
</tr>
<tr>
<td>tickCommand</td>
<td></td>
</tr>
<tr>
<td>unmount</td>
<td></td>
</tr>
<tr>
<td>unmountObject</td>
<td>(SceneObject target)</td>
</tr>
</tbody>
</table>
**TriggerData Member List**

This is the complete list of members for *TriggerData*, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>call</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>category</code></td>
<td>GameBaseData</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clientSide</code></td>
<td>TriggerData</td>
</tr>
<tr>
<td><code>clone</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getCanSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicField</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>onAdd(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onEnterTrigger(Trigger trigger, GameBase obj)</td>
<td>TriggerData</td>
</tr>
<tr>
<td>onLeaveTrigger(Trigger trigger, GameBase obj)</td>
<td>TriggerData</td>
</tr>
<tr>
<td>onMount(GameBase obj, SceneObject mountObj, int node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onNewDataBlock(GameBase obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td><code>onRemove</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onTickTrigger</code></td>
<td><code>TriggerData</code></td>
</tr>
<tr>
<td><code>onUnmount</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>reloadOnLocalClient</code></td>
<td><code>SimDataBlock</code></td>
</tr>
<tr>
<td><code>save</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCanSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>tickPeriodMS</code></td>
<td><code>TriggerData</code></td>
</tr>
</tbody>
</table>
# TSForestItemData Member List

This is the complete list of members for `TSForestItemData`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>branchAmp</td>
<td>ForestItemData</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>collidable</td>
<td>ForestItemData</td>
</tr>
<tr>
<td>dampingCoefficient</td>
<td>ForestItemData</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>detailAmp</td>
<td>ForestItemData</td>
</tr>
<tr>
<td>detailFreq</td>
<td>ForestItemData</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mass</td>
<td>ForestItemData</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>radius</td>
<td>ForestItemData</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>rigidity</td>
<td>ForestItemData</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>shapeFile</td>
<td>ForestItemData</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tightnessCoefficient</td>
<td>ForestItemData</td>
</tr>
<tr>
<td>trunkBendScale</td>
<td>ForestItemData</td>
</tr>
<tr>
<td>windScale</td>
<td>ForestItemData</td>
</tr>
</tbody>
</table>
# TSShapeConstructor Member List

This is the complete list of members for TSShapeConstructor, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>addCollisionDetail</code></td>
<td>Adds collision detail with specified parameters.</td>
</tr>
<tr>
<td><code>addImposter</code></td>
<td>Adds an imposter with specified parameters.</td>
</tr>
<tr>
<td><code>addMesh</code></td>
<td>Adds a mesh with specified parameters.</td>
</tr>
<tr>
<td><code>addNode</code></td>
<td>Adds a node with specified parameters.</td>
</tr>
<tr>
<td><code>addPrimitive</code></td>
<td>Adds a primitive with specified parameters.</td>
</tr>
<tr>
<td><code>addSequence</code></td>
<td>Adds a sequence with specified parameters.</td>
</tr>
<tr>
<td><code>addTrigger</code></td>
<td>Adds a trigger with specified parameters.</td>
</tr>
<tr>
<td><code>adjustCenter</code></td>
<td>Adjusts center for the shape.</td>
</tr>
<tr>
<td><code>adjustFloor</code></td>
<td>Adjusts floor for the shape.</td>
</tr>
<tr>
<td><code>alwaysImport</code></td>
<td>Always imports specified objects.</td>
</tr>
<tr>
<td><code>alwaysImportMesh</code></td>
<td>Always imports mesh objects.</td>
</tr>
<tr>
<td><code>assignFieldsFrom</code></td>
<td>Assigns fields from another object.</td>
</tr>
<tr>
<td><code>assignPersistentId</code></td>
<td>Assigns a persistent ID.</td>
</tr>
<tr>
<td><code>baseShape</code></td>
<td>Accesses base shape.</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>call</code> (string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dump</code> (bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpShape</code> (string filename='')</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>forceUpdateMaterials</code></td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getBounds()</code></td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getClassNamespace()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDetailLevelCount()</code></td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getDetailLevelIndex</code> (int size)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getDetailLevelName</code> (int index)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getDetailLevelSize</code> (int index)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getDynamicField</code> (int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getField</code> (int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>getFieldType</code> (string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue</code> (string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getImposterDetailLevel</code></td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getImposterSettings</code> (int index)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getInternalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getMeshCount</code> (string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getMeshMaterial</code> (string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getMeshName</code> (string name, int index)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getMeshSize</code> (string name, int index)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getMeshType</code> (string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getNodeChildCount</code> (string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getNodeChildName</code> (string name, int index)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getNodeCount</code> (string name, int index)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getNodeIndex</code> (string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getNodeName</code> (int index)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getNodeObjectName</code> (string name, int index)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getNodeParentName</code> (string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getNodeType</code> (string name, bool isWorld=false)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getObjectCount</code></td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getObjectIndex</code> (string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getObjectName</code> (int index)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>getObjectNode</code> (string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getSequenceBlend</code> (string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getSequenceCount</code> ()</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getSequenceCyclic</code> (string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getSequenceFrameCount</code> (string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getSequenceGroundSpeed</code> (string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getSequenceIndex</code> (string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getSequenceName</code> (int index)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getSequencePriority</code> (string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getSequenceSource</code> (string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getTargetCount</code> ()</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getTargetName</code> (int index)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getTrigger</code> (string name, int index)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>getTriggerCount</code> (string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>ignoreNodeScale</code></td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isChildOfGroup</code> (SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEditorOnly</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isExpanded</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField</code> (string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass</code> (string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod</code> (string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isSelected</code> ()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method/Field</td>
<td>Type</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>lodType</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>matNamePrefix</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>neverImport</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>neverImportMesh</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>notifyShapeChanged()</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>onLoad()</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>onUnload()</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>removeDetailLevel(int index)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>removeImposter()</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>removeMesh(string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>removeNode(string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>removeObject(string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>removeSequence(string name)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>removeTrigger(string name, int keyframe, int state)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>renameDetailLevel(string oldName, string newName)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>renameNode(string oldName, string newName)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>renameObject(string oldName, string newName)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>renameSequence(string oldName, string newName)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>saveShape(string filename)</td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>schedule</td>
<td>schedule(float time, string method, string args...)</td>
</tr>
<tr>
<td>sequence</td>
<td>sequence</td>
</tr>
<tr>
<td>setBounds</td>
<td>setBounds(Box3F bbox)</td>
</tr>
<tr>
<td>setCanSave</td>
<td>setCanSave(bool value=true)</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>setClassNamespace(string name)</td>
</tr>
<tr>
<td>setDetailLevelSize</td>
<td>setDetailLevelSize(int index, int newSize)</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>setEditorOnly(bool value=true)</td>
</tr>
<tr>
<td>setFieldType</td>
<td>setFieldType(string fieldName, string type)</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
</tr>
<tr>
<td>setFilename</td>
<td>setFilename(string fileName)</td>
</tr>
<tr>
<td>setHidden</td>
<td>setHidden(bool value=true)</td>
</tr>
<tr>
<td>setInternalName</td>
<td>setInternalName(string newInternalName)</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>setIsExpanded(bool state=true)</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>setIsSelected(bool state=true)</td>
</tr>
<tr>
<td>setLocked</td>
<td>setLocked(bool value=true)</td>
</tr>
<tr>
<td>setMeshMaterial</td>
<td>setMeshMaterial(string meshName, string matName)</td>
</tr>
<tr>
<td>setMeshSize</td>
<td>setMeshSize(string name, int size)</td>
</tr>
<tr>
<td>setMeshType</td>
<td>setMeshType(string name, string type)</td>
</tr>
<tr>
<td>setName</td>
<td>setName(string newName)</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>setNameChangeAllowed(bool value=true)</td>
</tr>
<tr>
<td>setNodeParent</td>
<td>setNodeParent(string name, string parentName)</td>
</tr>
<tr>
<td>setNodeTransform</td>
<td>setNodeTransform(string name, TransformF txfm, bool isWorld=false)</td>
</tr>
<tr>
<td>setObjectNode</td>
<td>setObjectNode(string objName, string nodeName)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>setSequenceBlend(string name, bool blend, string blendSeq, int blendFrame)</code></td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>setSequenceCyclic(string name, bool cyclic)</code></td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>setSequenceGroundSpeed(string name, Point3F transSpeed, Point3F rotSpeed=Point3F::Zero)</code></td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>setSequencePriority(string name, float priority)</code></td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>singleDetailSize</code></td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>unit</code></td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>upAxis</code></td>
<td>TSShapeConstructor</td>
</tr>
<tr>
<td><code>writeChangeSet()</code></td>
<td>TSShapeConstructor</td>
</tr>
</tbody>
</table>
## TSStatic Member List

This is the complete list of members for **TSStatic**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>allowPlayerStep</td>
<td>TSStatic</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>changeMaterial(string mapTo=&quot;&quot;, Material oldMat=NULL, Material newMat=NULL)</td>
<td>TSStatic</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>collisionType</td>
<td>TSStatic</td>
</tr>
<tr>
<td>decalType</td>
<td>TSStatic</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>forceDetail</td>
<td>TSStatic</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Name</td>
<td>Return Type</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getModelFile()</td>
<td>TSStatic</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTargetCount()</td>
<td>TSStatic</td>
</tr>
<tr>
<td>getTargetName(int index=0)</td>
<td>TSStatic</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>TSStatic [static]</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>TSStatic [static]</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>meshCulling</td>
<td>TSStatic</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>originSort</td>
<td>TSStatic</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAmbient</td>
<td>TSStatic</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>renderNormals</td>
<td>TSStatic</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>shapeName</td>
<td>TSStatic</td>
</tr>
<tr>
<td>skin</td>
<td>TSStatic</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
</tbody>
</table>
# TurretShape Member List

This is the complete list of members for `TurretShape`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>applyDamage(float amount)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>applyImpulse(Point3F pos, Point3F vec)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>GameBase::applyImpulse(Point3F pos, VectorF vel)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>applyRadialImpulse(Point3F origin, float radius, float magnitude)</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>applyRepair(float amount)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>boundingBox</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canCloak()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>changeMaterial(string mapTo, Material oldMat, Material newMat)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dataBlock</code></td>
<td>GameBase</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>destroyThread(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>doRespawn()</code></td>
<td>TurretShape</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMeshVisibility()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getAIRepairPoint()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getAllowManualFire()</code></td>
<td><code>TurretShape</code></td>
</tr>
<tr>
<td><code>getAllowManualRotation()</code></td>
<td><code>TurretShape</code></td>
</tr>
<tr>
<td><code>getCameraFov()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getCanSave()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespaces()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getControllingClient()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getControllingObject()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamageFlash()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamageLevel()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamagePercent()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDamageState()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDataBlock()</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>getDebugInfo()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDefaultCameraFov()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getDynamicField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getEnergyLevel()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getEnergyPercent()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getEulerRotation()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>getEyePoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeTransform()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeVector()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getImageAltTrigger(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageAmmo(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageGenericTrigger(int slot, int trigger)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageLoaded(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageScriptAnimPrefix(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageSkinTag(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageState(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageTarget(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageTrigger(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getLastStickyNormal()</td>
<td>Item</td>
</tr>
<tr>
<td>getLastStickyPos()</td>
<td>Item</td>
</tr>
<tr>
<td>getLookAtPoint(float distance=2000, int typeMask=0xFFFFFFFF)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>getMaxDamage()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getModelFile()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountSlot(ShapeBaseImageData)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzlePoint(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzleVector(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPendingImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getPosition(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRechargeRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRepairRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getShapeName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSkinName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSlotTransform(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getState()</td>
<td>TurretShape</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTargetCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getTargetName(int index)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getTurretEulerRotation()</code></td>
<td><code>TurretShape</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getVelocity()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getWhiteOut()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hasImageState(int slot, string state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isAIControlled</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isAtRest()</code></td>
<td><code>Item</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isCloaked()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDestroyed()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDisabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEnabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isHidden()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageFiring(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageMounted(ShapeBaseImageData image)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method/Property</td>
<td>Type</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>TurretShape</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isRotating()</td>
<td>Item</td>
</tr>
<tr>
<td>isSelectable</td>
<td>TurretShape</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isStatic()</td>
<td>Item</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>maxWarpTicks</td>
<td>Item</td>
</tr>
<tr>
<td>minWarpTicks</td>
<td>Item</td>
</tr>
<tr>
<td>mountImage(ShapeBaseImageData image, int slot, bool loaded=true, string skinTag=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SceneObject</td>
</tr>
<tr>
<td>onEnterLiquid(string objID, string waterCoverage, string liquidType)</td>
<td>Item</td>
</tr>
<tr>
<td>onLeaveLiquid(string objID, string liquidType)</td>
<td>Item</td>
</tr>
<tr>
<td>onStickyCollision(string objID)</td>
<td>Item</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pauseThread (int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAudio (int slot, SFXTrack track)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>playThread (int slot, string name=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>respawn</td>
<td>TurretShape</td>
</tr>
<tr>
<td>rotate</td>
<td>Item</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save (string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule (float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient (NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setAllMeshesHidden (bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setAllowManualFire (bool allow)</td>
<td>TurretShape</td>
</tr>
<tr>
<td>setAllowManualRotation (bool allow)</td>
<td>TurretShape</td>
</tr>
<tr>
<td>setCameraFov (float fov)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCanSave (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCloaked (bool cloak)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCollisionTimeout (int ignoreColObj=NULL)</td>
<td>Item</td>
</tr>
<tr>
<td>setControl (bool controlled)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setDamageFlash (float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageLevel (float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageState (string state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>setDamageVector(Point3F vec)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setDataBlock(GameBaseData data)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEnergyLevel(float level)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden(bool show)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageAltTrigger(int slot, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageAmmo(int slot, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageGenericTrigger(int slot, int trigger, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageLoaded(int slot, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageScriptAnimPrefix(int slot, string prefix)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageTarget(int slot, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageTrigger(int slot, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInvincibleMode(float time, float speed)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setMeshHidden(string name, bool hide)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setRechargeRate(float rate)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setRepairRate(float rate)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setShapeName(string name)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setSkinName(string name)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setThreadDir(int slot, bool fwd)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setThreadPosition(int slot, float pos)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setThreadTimeScale(int slot, float scale)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setTurretEulerRotation(Point3F rot)</code></td>
<td>TurretShape</td>
</tr>
<tr>
<td><code>setVelocity(Point3F vel)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>setWhiteOut(float level)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>skin</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>startFade(int time, int delay, bool fadeOut)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>static</code></td>
<td>Item</td>
</tr>
<tr>
<td><code>stopAudio(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>stopThread(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>unmountImage(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>validateCameraFov(float fov)</code></td>
<td>ShapeBase</td>
</tr>
</tbody>
</table>
# TurretShapeData Member List

This is the complete list of members for `TurretShapeData`, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>cameraDefaultFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMaxDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMaxFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraOffset</code></td>
<td><code>TurretShapeData</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>category</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>checkDeployPos(TransformF txfm)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>computeCRC</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cubeReflectorDesc</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>Debris</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>debrisShapeName</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>density</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>destroyedLevel</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>disabledLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>drag</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>elasticity</td>
<td>ItemData</td>
</tr>
<tr>
<td>Explosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>firstPersonOnly</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>friction</td>
<td>ItemData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeployTransform(Point3F pos, Point3F normal)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>fieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getld()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>gravityMod</td>
<td>ItemData</td>
</tr>
<tr>
<td>headingRate</td>
<td>TurretShapeData</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>inheritEnergyFromMount</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInvincible</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>lightColor</td>
<td>ItemData</td>
</tr>
<tr>
<td>lightOnlyStatic</td>
<td>ItemData</td>
</tr>
<tr>
<td>lightRadius</td>
<td>ItemData</td>
</tr>
<tr>
<td>lightTime</td>
<td>ItemData</td>
</tr>
<tr>
<td>lightType</td>
<td>ItemData</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mass</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>maxDamage</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>maxEnergy</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>maxHeading</td>
<td>TurretShapeData</td>
</tr>
<tr>
<td>maxPitch</td>
<td>TurretShapeData</td>
</tr>
<tr>
<td>maxVelocity</td>
<td>ItemData</td>
</tr>
<tr>
<td>minPitch</td>
<td>TurretShapeData</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>observeThroughObject</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onAdd(obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onCollision(obj, collObj, vec, len)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onDamage(obj, delta)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onDestroyed(obj, lastState)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onDisabled(obj, lastState)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onEnabled(obj, lastState)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onEndSequence(obj, slot)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onForceUncloak(obj, reason)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onImpact(obj, collObj, vec, len)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onMount(obj, mountObj, node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onMountObject(turret, obj, node)</td>
<td>TurretShapeData</td>
</tr>
<tr>
<td>onNewDataBlock(obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onRemove(obj)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onStickyCollision(obj)</td>
<td>TurretShapeData</td>
</tr>
<tr>
<td>onTrigger(obj, index, state)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onUnmount(obj, mountObj, node)</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>onUnmountObject(turret, obj)</td>
<td>TurretShapeData</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>pitchRate</td>
<td>TurretShapeData</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>renderWhenDestroyed</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>repairRate</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>shadowEnable</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowMaxVisibleDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowProjectionDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSize</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSphereAdjust</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shapeFile</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>simpleServerCollision</td>
<td>ItemData</td>
</tr>
<tr>
<td>startLoaded</td>
<td>TurretShapeData</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>sticky</td>
<td>ItemData</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>underwaterExplosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>useEyePoint</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>weaponLinkType</td>
<td>TurretShapeData</td>
</tr>
<tr>
<td>zRotOnly</td>
<td>TurretShapeData</td>
</tr>
</tbody>
</table>
Vehicle Member List

This is the complete list of members for **Vehicle**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>applyDamage(float amount)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>applyImpulse(Point3F pos, Point3F vec)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>GameBase::applyImpulse(Point3F pos, VectorF vel)</td>
<td>GameBase</td>
</tr>
<tr>
<td>applyRadialImpulse(Point3F origin, float radius, float magnitude)</td>
<td>GameBase</td>
</tr>
<tr>
<td>applyRepair(float amount)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>boundingBox</td>
<td>GameBase [static]</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canCloak()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>changeMaterial(string mapTo, Material oldMat, Material newMat)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dataBlock</td>
<td>GameBase</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>destroyThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>disableMove</td>
<td>Vehicle</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMeshVisibility()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getAIRepairPoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getControllingClient()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getControllingObject()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageFlash()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamagePercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageState()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDefaultCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEnergyLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEnergyPercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getEyePoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeTransform()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>getEyeVector()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getImageAltTrigger(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageAmmo(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageGenericTrigger(int slot, int trigger)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageLoaded(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageScriptAnimPrefix(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageSkinTag(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageState(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageTarget(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageTrigger(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getLookAtPoint(float distance=2000, int typeMask=0xFFFFFFFF)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMaxDamage()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getModelFile()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountSlot(ShapeBaseImageData)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzlePoint(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzleVector(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPendingImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRechargeRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRepairRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getShapeName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSkinName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSlotTransform(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTargetCount()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTargetName(int index)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getVelocity()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getWhiteOut()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>--------------------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>hasImageState(int slot, string state)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>hidden</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isAIControlled</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>isCloaked()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isDestroyed()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isDisabled()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isEnabled()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isHidden()</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isImageFiring(int slot)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isImage Mounted(ShapeBaseImageData image)</code></td>
<td>ShapeBase</td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td>Vehicle</td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td>Vehicle</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountImage(image, slot, loaded=true, skinTag='')</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(objB, slot, txfm=Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pauseThread(slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAudio(slot, track)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>playThread(slot, name='')</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(fileName, selectedOnly=false, preAppendString='')</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(time, method, args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setAllMeshesHidden(hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCameraFov(fov)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCloaked(bool cloak)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setControl(bool controlled)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>setDamageFlash(float level)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setDamageLevel(float level)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setDamageState(string state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setDamageVector(Point3F vec)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setDataBlock(GameBaseData data)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEnergyLevel(float level)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code> setHidden(bool show)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageAltTrigger(int slot, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageAmmo(int slot, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageGenericTrigger(int slot, int trigger, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageLoaded(int slot, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageScriptAnimPrefix(int slot, string prefix)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageTarget(int slot, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageTrigger(int slot, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInvincibleMode(float time, float speed)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setMeshHidden</code> (string name, bool hide)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setRechargeRate</code> (float rate)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setRepairRate</code> (float rate)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setScale</code> (Point3F scale)</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways</code> ()</td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setShapeName</code> (string name)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setSkinName</code> (string name)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setThreadDir</code> (int slot, bool fwd)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setThreadPosition</code> (int slot, float pos)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setThreadTimeScale</code> (int slot, float scale)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setTransform</code> (TransformF txfm)</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setVelocity</code> (Point3F vel)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setWhiteOut</code> (float level)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>skin</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>startFade</code> (int time, int delay, bool fadeOut)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>stopAudio</code> (int slot)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>stopThread</code> (int slot)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>unmount</code> ()</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountImage</code> (int slot)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>unmountObject</code> (SceneObject target)</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>validateCameraFov</code> (float fov)</td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>workingQueryBoxSizeMultiplier</code></td>
<td><code>Vehicle</code></td>
</tr>
<tr>
<td></td>
<td><code>[static]</code></td>
</tr>
<tr>
<td><code>workingQueryBoxStaleThreshold</code></td>
<td><code>Vehicle</code></td>
</tr>
<tr>
<td></td>
<td><code>[static]</code></td>
</tr>
</tbody>
</table>
Copyright © GarageGames, LLC. All Rights Reserved.
# VehicleData Member List

This is the complete list of members for VehicleData, including all inherited members.

<table>
<thead>
<tr>
<th>Method/Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>bodyFriction</td>
<td>VehicleData</td>
</tr>
<tr>
<td>bodyRestitution</td>
<td>VehicleData</td>
</tr>
<tr>
<td>call</td>
<td>SimObject</td>
</tr>
<tr>
<td>cameraDecay</td>
<td>VehicleData</td>
</tr>
<tr>
<td>cameraDefaultFov</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cameraLag</td>
<td>VehicleData</td>
</tr>
<tr>
<td>cameraMaxDist</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cameraMaxFov</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cameraMinDist</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cameraMinFov</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>cameraOffset</td>
<td>VehicleData</td>
</tr>
<tr>
<td>cameraRoll</td>
<td>VehicleData</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>category</td>
<td>GameBaseData</td>
</tr>
<tr>
<td>checkDeployPos</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clone</td>
<td>SimObject</td>
</tr>
<tr>
<td>collDamageMultiplier</td>
<td>VehicleData</td>
</tr>
<tr>
<td>collDamageThresholdVel</td>
<td>VehicleData</td>
</tr>
<tr>
<td>collisionTol</td>
<td>VehicleData</td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>computeCRC</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>contactTol</td>
<td>VehicleData</td>
</tr>
<tr>
<td>cubeReflectorDesc</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>damageEmitter</td>
<td>VehicleData</td>
</tr>
<tr>
<td>damageEmitterOffset</td>
<td>VehicleData</td>
</tr>
<tr>
<td>damageLevelTolerance</td>
<td>VehicleData</td>
</tr>
<tr>
<td>Debris</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>debrisShapeName</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>density</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>destroyedLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>disabledLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>drag</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dustEmitter</td>
<td>VehicleData</td>
</tr>
<tr>
<td>dustHeight</td>
<td>VehicleData</td>
</tr>
<tr>
<td>exitingWater</td>
<td>VehicleData</td>
</tr>
<tr>
<td>exitSplashSoundVelocity</td>
<td>VehicleData</td>
</tr>
<tr>
<td>Explosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>firstPersonOnly</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeployTransform(Point3F pos, Point3F normal)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hardImpactSound</td>
<td>VehicleData</td>
</tr>
<tr>
<td>hardImpactSpeed</td>
<td>VehicleData</td>
</tr>
<tr>
<td>hardSplashSoundVelocity</td>
<td>VehicleData</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>impactWaterEasy</td>
<td>VehicleData</td>
</tr>
<tr>
<td>impactWaterHard</td>
<td>VehicleData</td>
</tr>
<tr>
<td>impactWaterMedium</td>
<td>VehicleData</td>
</tr>
<tr>
<td>inheritEnergyFromMount</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>integration</td>
<td>VehicleData</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method/Property</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInvincible</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>jetEnergyDrain</td>
<td>VehicleData</td>
</tr>
<tr>
<td>jetForce</td>
<td>VehicleData</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mass</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>massBox</td>
<td>VehicleData</td>
</tr>
<tr>
<td>massCenter</td>
<td>VehicleData</td>
</tr>
<tr>
<td>maxDamage</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>maxDrag</td>
<td>VehicleData</td>
</tr>
<tr>
<td>maxEnergy</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>maxSteeringAngle</td>
<td>VehicleData</td>
</tr>
<tr>
<td>mediumSplashSoundVelocity</td>
<td>VehicleData</td>
</tr>
<tr>
<td>minDrag</td>
<td>VehicleData</td>
</tr>
<tr>
<td>minImpactSpeed</td>
<td>VehicleData</td>
</tr>
<tr>
<td>minJetEnergy</td>
<td>VehicleData</td>
</tr>
<tr>
<td>minRollSpeed</td>
<td>VehicleData</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>numDmgEmitterAreas</td>
<td>VehicleData</td>
</tr>
<tr>
<td>observeThroughObject</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onAdd(GameBase obj)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>onCollision(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>onDamage</strong></td>
<td>(ShapeBase obj, float delta)</td>
</tr>
<tr>
<td><strong>onDestroyed</strong></td>
<td>(ShapeBase obj, string lastState)</td>
</tr>
<tr>
<td><strong>onDisabled</strong></td>
<td>(ShapeBase obj, string lastState)</td>
</tr>
<tr>
<td><strong>onEnabled</strong></td>
<td>(ShapeBase obj, string lastState)</td>
</tr>
<tr>
<td><strong>onEndSequence</strong></td>
<td>(ShapeBase obj, int slot)</td>
</tr>
<tr>
<td><strong>onEnterLiquid</strong></td>
<td>(Vehicle obj, float coverage, string type)</td>
</tr>
<tr>
<td><strong>onForceUncloak</strong></td>
<td>(ShapeBase obj, string reason)</td>
</tr>
<tr>
<td><strong>onImpact</strong></td>
<td>(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</td>
</tr>
<tr>
<td><strong>onLeaveLiquid</strong></td>
<td>(Vehicle obj, string type)</td>
</tr>
<tr>
<td><strong>onMount</strong></td>
<td>(GameBase obj, SceneObject mountObj, int node)</td>
</tr>
<tr>
<td><strong>onNewDataBlock</strong></td>
<td>(GameBase obj)</td>
</tr>
<tr>
<td><strong>onRemove</strong></td>
<td>(GameBase obj)</td>
</tr>
<tr>
<td><strong>onTrigger</strong></td>
<td>(ShapeBase obj, int index, bool state)</td>
</tr>
<tr>
<td><strong>onUnmount</strong></td>
<td>(GameBase obj, SceneObject mountObj, int node)</td>
</tr>
<tr>
<td><strong>parentGroup</strong></td>
<td></td>
</tr>
<tr>
<td><strong>persistentId</strong></td>
<td></td>
</tr>
<tr>
<td><strong>reloadOnLocalClient</strong></td>
<td>()</td>
</tr>
<tr>
<td><strong>renderWhenDestroyed</strong></td>
<td></td>
</tr>
<tr>
<td><strong>repairRate</strong></td>
<td></td>
</tr>
<tr>
<td><strong>save</strong></td>
<td>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
</tr>
<tr>
<td><strong>schedule</strong></td>
<td>(float time, string method, string args...)</td>
</tr>
<tr>
<td><strong>setCanSave</strong></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td><strong>setClassNamespace</strong></td>
<td>(string name)</td>
</tr>
<tr>
<td><strong>setEditorOnly</strong></td>
<td>(bool value=true)</td>
</tr>
<tr>
<td>Function Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>setFieldType</code></td>
<td>set field type</td>
</tr>
<tr>
<td><code>setFieldValue</code></td>
<td>set field value</td>
</tr>
<tr>
<td><code>setFilename</code></td>
<td>set filename</td>
</tr>
<tr>
<td><code>setHidden</code></td>
<td>set hidden</td>
</tr>
<tr>
<td><code>setInternalName</code></td>
<td>set internal name</td>
</tr>
<tr>
<td><code>setIsExpanded</code></td>
<td>set is expanded</td>
</tr>
<tr>
<td><code>setIsSelected</code></td>
<td>set is selected</td>
</tr>
<tr>
<td><code>setLocked</code></td>
<td>set locked</td>
</tr>
<tr>
<td><code>setName</code></td>
<td>set name</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code></td>
<td>set name change allowed</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code></td>
<td>set super class namespace</td>
</tr>
<tr>
<td><code>shadowEnable</code></td>
<td>enable shadow</td>
</tr>
<tr>
<td><code>shadowMaxVisibleDistance</code></td>
<td>max visible distance</td>
</tr>
<tr>
<td><code>shadowProjectionDistance</code></td>
<td>projection distance</td>
</tr>
<tr>
<td><code>shadowSize</code></td>
<td>size</td>
</tr>
<tr>
<td><code>shadowSphereAdjust</code></td>
<td>sphere adjust</td>
</tr>
<tr>
<td><code>shapeFile</code></td>
<td>shape file</td>
</tr>
<tr>
<td><code>softImpactSound</code></td>
<td>soft impact sound</td>
</tr>
<tr>
<td><code>softImpactSpeed</code></td>
<td>soft impact speed</td>
</tr>
<tr>
<td><code>softSplashSoundVelocity</code></td>
<td>soft splash sound velocity</td>
</tr>
<tr>
<td><code>splashEmitter</code></td>
<td>splash emitter</td>
</tr>
<tr>
<td><code>splashFreqMod</code></td>
<td>splash freq mod</td>
</tr>
<tr>
<td><code>splashVelEpsilon</code></td>
<td>splash velocity epsilon</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>super class</td>
</tr>
<tr>
<td><code>triggerDustHeight</code></td>
<td>trigger dust height</td>
</tr>
<tr>
<td><code>underwaterExplosion</code></td>
<td>underwater explosion</td>
</tr>
<tr>
<td><code>useEyePoint</code></td>
<td>use eye point</td>
</tr>
</tbody>
</table>
# WaterBlock Member List

This is the complete list of members for **WaterBlock**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>baseColor</td>
<td>WaterObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>clarity</td>
<td>WaterObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>cubemap</td>
<td>WaterObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>density</td>
<td>WaterObject</td>
</tr>
<tr>
<td>depthGradientMax</td>
<td>WaterObject</td>
</tr>
<tr>
<td>depthGradientTex</td>
<td>WaterObject</td>
</tr>
<tr>
<td>distortEndDist</td>
<td>WaterObject</td>
</tr>
<tr>
<td>distortFullDepth</td>
<td>WaterObject</td>
</tr>
<tr>
<td>distortStartDist</td>
<td>WaterObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Return Type</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>emissive</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamAmbientLerp</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamDir</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamMaxDepth</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamOpacity</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamRippleInfluence</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamSpeed</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamTex</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamTexScale</td>
<td>WaterObject</td>
</tr>
<tr>
<td>fresnelBias</td>
<td>WaterObject</td>
</tr>
<tr>
<td>fresnelPower</td>
<td>WaterObject</td>
</tr>
<tr>
<td>fullReflect</td>
<td>WaterObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>gridElementSize</td>
<td>WaterBlock</td>
</tr>
<tr>
<td>gridSize</td>
<td>WaterBlock</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>WaterBlock</td>
</tr>
<tr>
<td>[static]</td>
<td></td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>WaterBlock</td>
</tr>
<tr>
<td>[static]</td>
<td></td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>liquidType</td>
<td>WaterObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SceneObject</td>
</tr>
<tr>
<td>overallFoamOpacity</td>
<td>WaterObject</td>
</tr>
<tr>
<td>overallRippleMagnitude</td>
<td>WaterObject</td>
</tr>
<tr>
<td>overallWaveMagnitude</td>
<td>WaterObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>reflectDetailAdjust</td>
<td>WaterObject</td>
</tr>
<tr>
<td>reflectivity</td>
<td>WaterObject</td>
</tr>
<tr>
<td>reflectMaxRateMs</td>
<td>WaterObject</td>
</tr>
<tr>
<td>reflectNormalUp</td>
<td>WaterObject</td>
</tr>
<tr>
<td>reflectPriority</td>
<td>WaterObject</td>
</tr>
<tr>
<td>reflectTexSize</td>
<td>WaterObject</td>
</tr>
<tr>
<td>rippleDir</td>
<td>WaterObject</td>
</tr>
<tr>
<td>rippleMagnitude</td>
<td>WaterObject</td>
</tr>
<tr>
<td>rippleSpeed</td>
<td>WaterObject</td>
</tr>
<tr>
<td>rippleTex</td>
<td>WaterObject</td>
</tr>
<tr>
<td>rippleTexScale</td>
<td>WaterObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td><strong>save</strong>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td><strong>schedule</strong>(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>soundAmbience</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>specularColor</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>specularPower</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>underwaterColor</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>useOcclusionQuery</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>viscosity</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>waterFogDensity</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>waterFogDensityOffset</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>waveDir</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>waveMagnitude</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>waveSpeed</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>wetDarkening</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>wetDepth</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>wireframe</code></td>
<td><code>WaterObject</code></td>
</tr>
</tbody>
</table>

[static]
# WaterObject Member List

This is the complete list of members for **WaterObject**, including all inherited members.

<table>
<thead>
<tr>
<th>Method/Function</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>baseColor</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSave</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clarity</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>class</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>className</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>clearScopeToClient(NetConnection client)</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>clone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>cubemap</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>deepClone()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>delete()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>density</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>depthGradientMax</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>depthGradientTex</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>distortEndDist</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>distortFullDepth</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>distortStartDist</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>dump(bool detailed=false)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpClassHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpGroupHierarchy()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>dumpMethods()</code></td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Container</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>emissive</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamAmbientLerp</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamDir</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamMaxDepth</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamOpacity</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamRippleInfluence</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamSpeed</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamTex</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamTexScale</td>
<td>WaterObject</td>
</tr>
<tr>
<td>fresnelBias</td>
<td>WaterObject</td>
</tr>
<tr>
<td>fresnelPower</td>
<td>WaterObject</td>
</tr>
<tr>
<td>fullReflect</td>
<td>WaterObject</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><code>isField</code> (string fieldName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass</code> (string className)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod</code> (string methodName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td></td>
<td>[static]</td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td></td>
<td>[static]</td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelectionEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>liquidType</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>mountNode</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountObject</code> (SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountPID</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountPos</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>mountRot</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>overallFoamOpacity</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>overallRippleMagnitude</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>overallWaveMagnitude</code></td>
<td><code>WaterObject</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>position</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>reflectDetailAdjust</td>
<td>WaterObject</td>
</tr>
<tr>
<td>reflectivity</td>
<td>WaterObject</td>
</tr>
<tr>
<td>reflectMaxRateMs</td>
<td>WaterObject</td>
</tr>
<tr>
<td>reflectNormalUp</td>
<td>WaterObject</td>
</tr>
<tr>
<td>reflectPriority</td>
<td>WaterObject</td>
</tr>
<tr>
<td>reflectTexSize</td>
<td>WaterObject</td>
</tr>
<tr>
<td>rippleDir</td>
<td>WaterObject</td>
</tr>
<tr>
<td>rippleMagnitude</td>
<td>WaterObject</td>
</tr>
<tr>
<td>rippleSpeed</td>
<td>WaterObject</td>
</tr>
<tr>
<td>rippleTex</td>
<td>WaterObject</td>
</tr>
<tr>
<td>rippleTexScale</td>
<td>WaterObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setScale(Point3F scale)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setScopeAlways()</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setTransform(TransformF txfm)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>soundAmbience</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>specularColor</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>specularPower</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>underwaterColor</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>unmount()</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>unmountObject(SceneObject target)</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>useOcclusionQuery</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>viscosity</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>waterFogDensity</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>waterFogDensityOffset</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>waveDir</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>waveMagnitude</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>waveSpeed</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>wetDarkening</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>wetDepth</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>wireframe</code></td>
<td>WaterObject</td>
</tr>
</tbody>
</table>

[static]
**WaterPlane Member List**

This is the complete list of members for *WaterPlane*, including all inherited members.

<table>
<thead>
<tr>
<th>Member Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>baseColor</td>
<td>WaterObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>clarity</td>
<td>WaterObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>cubemap</td>
<td>WaterObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>density</td>
<td>WaterObject</td>
</tr>
<tr>
<td>depthGradientMax</td>
<td>WaterObject</td>
</tr>
<tr>
<td>depthGradientTex</td>
<td>WaterObject</td>
</tr>
<tr>
<td>distortEndDist</td>
<td>WaterObject</td>
</tr>
<tr>
<td>distortFullDepth</td>
<td>WaterObject</td>
</tr>
<tr>
<td>distortStartDist</td>
<td>WaterObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>emissive</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamAmbientLerp</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamDir</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamMaxDepth</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamOpacity</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamRippleInfluence</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamSpeed</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamTex</td>
<td>WaterObject</td>
</tr>
<tr>
<td>foamTexScale</td>
<td>WaterObject</td>
</tr>
<tr>
<td>fresnelBias</td>
<td>WaterObject</td>
</tr>
<tr>
<td>fresnelPower</td>
<td>WaterObject</td>
</tr>
<tr>
<td>fullReflect</td>
<td>WaterObject</td>
</tr>
<tr>
<td>getCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEulerRotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------</td>
<td>---------------</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObject(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectCount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>gridElementSize</td>
<td>WaterPlane</td>
</tr>
<tr>
<td>gridSize</td>
<td>WaterPlane</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>WaterPlane</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>WaterPlane</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>liquidType</td>
<td>WaterObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SceneObject</td>
</tr>
<tr>
<td>overallFoamOpacity</td>
<td>WaterObject</td>
</tr>
<tr>
<td>overallRippleMagnitude</td>
<td>WaterObject</td>
</tr>
<tr>
<td>overallWaveMagnitude</td>
<td>WaterObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method/Property</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>reflectDetailAdjust</td>
<td>WaterObject</td>
</tr>
<tr>
<td>reflectivity</td>
<td>WaterObject</td>
</tr>
<tr>
<td>reflectMaxRateMs</td>
<td>WaterObject</td>
</tr>
<tr>
<td>reflectNormalUp</td>
<td>WaterObject</td>
</tr>
<tr>
<td>reflectPriority</td>
<td>WaterObject</td>
</tr>
<tr>
<td>reflectTexSize</td>
<td>WaterObject</td>
</tr>
<tr>
<td>rippleDir</td>
<td>WaterObject</td>
</tr>
<tr>
<td>rippleMagnitude</td>
<td>WaterObject</td>
</tr>
<tr>
<td>rippleSpeed</td>
<td>WaterObject</td>
</tr>
<tr>
<td>rippleTex</td>
<td>WaterObject</td>
</tr>
<tr>
<td>rippleTexScale</td>
<td>WaterObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>setInternalName</code> (string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected</code> (bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setLocked</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setScale</code> (Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>setScopeAlways</code></td>
<td>NetObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setTransform</code> (TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>soundAmbience</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>specularColor</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>specularPower</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>underwaterColor</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>unmount</code></td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>unmountObject</code> (SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>useOcclusionQuery</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>viscosity</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>waterFogDensity</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>waterFogDensityOffset</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>waveDir</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>waveMagnitude</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>waveSpeed</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>wetDarkening</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>wetDepth</code></td>
<td>WaterObject</td>
</tr>
<tr>
<td><code>wireframe</code></td>
<td>WaterObject</td>
</tr>
</tbody>
</table>
|                                             |                | [static]
## WayPoint Member List

This is the complete list of members for WayPoint, including all inherited members.

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>applyDamage</td>
<td>(float amount) ShapeBase</td>
</tr>
<tr>
<td>applyImpulse</td>
<td>(Point3F pos, Point3F vec) ShapeBase</td>
</tr>
<tr>
<td>GameBase::applyImpulse</td>
<td>(Point3F pos, VectorF vel) GameBase</td>
</tr>
<tr>
<td>applyRadialImpulse</td>
<td>(Point3F origin, float radius, float magnitude) GameBase</td>
</tr>
<tr>
<td>applyRepair</td>
<td>(float amount) ShapeBase</td>
</tr>
<tr>
<td>assignFieldsFrom</td>
<td>(SimObject fromObject) SimObject</td>
</tr>
<tr>
<td>assignPersistentId</td>
<td>() SimObject</td>
</tr>
<tr>
<td>boundingBox</td>
<td>GameBase [static]</td>
</tr>
<tr>
<td>call</td>
<td>(string method, string args...) SimObject</td>
</tr>
<tr>
<td>canCloak</td>
<td>() ShapeBase</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>changeMaterial</td>
<td>(string mapTo, Material oldMat, Material newMat) ShapeBase</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient</td>
<td>(NetConnection client) NetObject</td>
</tr>
<tr>
<td>clone</td>
<td>() SimObject</td>
</tr>
<tr>
<td>dataBlock</td>
<td>GameBase</td>
</tr>
<tr>
<td>deepClone</td>
<td>() SimObject</td>
</tr>
<tr>
<td>delete</td>
<td>() SimObject</td>
</tr>
<tr>
<td>destroyThread</td>
<td>(int slot) ShapeBase</td>
</tr>
<tr>
<td>dump</td>
<td>(bool detailed=false) SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMeshVisibility()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getAIRepairPoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getControllingClient()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getControllingObject()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageFlash()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamagePercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageState()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDefaultCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEnergyLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEnergyPercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getEyePoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeTransform()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeVector()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td><code>getField(int index)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getForwardVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getGhostID()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getImageAltTrigger(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageAmmo(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageGenericTrigger(int slot, int trigger)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageLoaded(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageScriptAnimPrefix(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageSkinTag(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageState(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageTarget(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getImageTrigger(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInverseTransform()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getLookAtPoint(float distance=2000, int typeMask=0xFFFFFFFF)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMaxDamage()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getModelFile()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMountedImage(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>getMountedObjectNode(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountNodeObject(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getMountSlot(ShapeBaseImageData)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzlePoint(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMuzzleVector(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getObjectBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getObjectMount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getPendingImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getPosition()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getRechargeRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRepairRate()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getRightVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getScale()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getShapeName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSkinName()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSlotTransform(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getTargetCount()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTargetName(int index)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getVelocity()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getWhiteOut()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hasImageState(int slot, string state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isAIControlled</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isCloaked()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDestroyed()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDisabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEnabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isHidden()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageFiring(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageMounted(ShapeBaseImageData image)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isRenderable</code></td>
<td><code>WayPoint</code></td>
</tr>
<tr>
<td><code>isRenderEnabled</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isSelectable</code></td>
<td><code>WayPoint</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function/Property</td>
<td>Class</td>
</tr>
<tr>
<td>------------------------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>markerName</td>
<td>WayPoint</td>
</tr>
<tr>
<td>mountImage(ShapeBaseImageData image, int slot, bool loaded=true, string skinTag=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF tfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pauseThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAudio(int slot, SFXTrack track)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>playThread(int slot, string name=&quot;&quot;)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>setAllMeshesHidden(bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class Name</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCloaked(bool cloak)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setControl(bool controlled)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setDamageFlash(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageLevel(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageState(string state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDamageVector(Point3F vec)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setDataBlock(GameBaseData data)</td>
<td>GameBase</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEnergyLevel(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool show)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAltTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageAmmo(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageGenericTrigger(int slot, int trigger, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageLoaded(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageScriptAnimPrefix(int slot, string prefix)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTarget(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setImageTrigger(int slot, bool state)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInvincibleMode(float time, float speed)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setMeshHidden(string name, bool hide)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setRechargeRate(float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setRepairRate(float rate)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setScale(Point3F scale)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setScopeAlways()</td>
<td>NetObject</td>
</tr>
<tr>
<td>setShapeName(string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setSkinName(string name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setThreadDir(int slot, bool fwd)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setThreadPosition(int slot, float pos)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setThreadTimeScale(int slot, float scale)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setTransform(TransformF txfm)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>setVelocity(Point3F vel)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>setWhiteOut(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>skin</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>startFade(int time, int delay, bool fadeOut)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>stopAudio(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>stopThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>team</td>
<td>WayPoint</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>validateCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
</tbody>
</table>
**WheeledVehicle Member List**

This is the complete list of members for *WheeledVehicle*, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>applyDamage(float amount)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>applyImpulse(Point3F pos, Point3F vec)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>GameBase::applyImpulse(Point3F pos, VectorF vel)</td>
<td>GameBase</td>
</tr>
<tr>
<td>applyRadialImpulse(Point3F origin, float radius, float magnitude)</td>
<td>GameBase</td>
</tr>
<tr>
<td>applyRepair(float amount)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>boundingBox</td>
<td>GameBase</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canCloak()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>changeMaterial(string mapTo, Material oldMat, Material newMat)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dataBlock</td>
<td>GameBase</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>destroyThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>disableMove</td>
<td>Vehicle</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMeshVisibility()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getAIRepairPoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getControllingClient()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getControllingObject()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageFlash()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamagePercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDamageState()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDataBlock()</td>
<td>GameBase</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDefaultCameraFov()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getEnergyLevel()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEnergyPercent()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEulerRotation()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>getEyePoint()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeTransform()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getEyeVector()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getForwardVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getGhostID()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getAltTrigger(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageAmmo(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageGenericTrigger(int slot, int trigger)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageLoaded(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageScriptAnimPrefix(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageSkinTag(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageState(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageTarget(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getImageTrigger(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInverseTransform()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getLookAtPoint(float distance=2000, int typeMask=0xFFFFFFF)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>getMaxDamage()</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><code>getModelFile()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMountedImage(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMountedObject(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectCount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountedObjectNode(int slot)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountNodeObject(int node)</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getMountSlot(ShapeBaseImageData image)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMuzzlePoint(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getMuzzleVector(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getObjectBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getObjectMount()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getPendingImage(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getPosition()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getRechargeRate()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getRepairRate()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getRightVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getScale()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getServerObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>getShapeName()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getSkinName()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getSlotTransform(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getTargetCount()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getTargetName(int index)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getTransform()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getType()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><code>getUpVector()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getVelocity()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getWheelCount()</code></td>
<td><code>WheeledVehicle</code></td>
</tr>
<tr>
<td><code>getWhiteOut()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>getWorldBox()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>getWorldBoxCenter()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>hasImageState(int slot, string state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>hidden(int slot, string state)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isAIControlled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isClientObject()</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>isCloaked()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDestroyed()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isDisabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEnabled()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isGlobalBounds()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>isHidden()</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageFiring(int slot)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isImageMounted(ShapeBaseImageData image)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMounted()</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>WheeledVehicle</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>WheeledVehicle</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountImage(image, slot, loaded, skinTag)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>mountNode</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountObject(objB, slot, txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>pauseThread(slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>persistentId(int slot)</td>
<td>SimObject</td>
</tr>
<tr>
<td>playAudio(slot, track)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>playThread(slot, name)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save(fileName, selectedOnly=false, preAppendString)</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method Call</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>scopeToClient(NetConnection client)</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setAllMeshesHidden(bool hide)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setCameraFov(float fov)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCloaked(bool cloak)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setControl(bool controlled)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>setDamageFlash(float level)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setDamageLevel(float level)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setDamageState(string state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setDamageVector(Point3F vec)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setDataBlock(GameBaseData data)</code></td>
<td><code>GameBase</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEnergyLevel(float level)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code> setHidden(bool show)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageAltTrigger(int slot, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageAmmo(int slot, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageGenericTrigger(int slot, int trigger, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageLoaded(int slot, bool state)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td><code>setImageScriptAnimPrefix(int slot, string prefix)</code></td>
<td><code>ShapeBase</code></td>
</tr>
<tr>
<td>Method Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>setImageTarget</td>
<td>Set image target</td>
</tr>
<tr>
<td>setImageTrigger</td>
<td>Set image trigger</td>
</tr>
<tr>
<td>setInternalName</td>
<td>Set internal name</td>
</tr>
<tr>
<td>setInvincibleMode</td>
<td>Set invincible mode</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>Set is expanded</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>Set is selected</td>
</tr>
<tr>
<td>setLocked</td>
<td>Set locked</td>
</tr>
<tr>
<td>setMeshHidden</td>
<td>Set mesh hidden</td>
</tr>
<tr>
<td>setName</td>
<td>Set name</td>
</tr>
<tr>
<td>setNameChangeAllowed</td>
<td>Set name change allowed</td>
</tr>
<tr>
<td>setRechargeRate</td>
<td>Set recharge rate</td>
</tr>
<tr>
<td>setRepairRate</td>
<td>Set repair rate</td>
</tr>
<tr>
<td>setScale</td>
<td>Set scale</td>
</tr>
<tr>
<td>setScopeAlways</td>
<td>Set scope always</td>
</tr>
<tr>
<td>setNameChange</td>
<td>Set name change</td>
</tr>
<tr>
<td>setSuperClassNamespace</td>
<td>Set super class namespace</td>
</tr>
<tr>
<td>setThreadDir</td>
<td>Set thread direction</td>
</tr>
<tr>
<td>setThreadPosition</td>
<td>Set thread position</td>
</tr>
<tr>
<td>setThreadTimeScale</td>
<td>Set thread time scale</td>
</tr>
<tr>
<td>setTransform</td>
<td>Set transform</td>
</tr>
<tr>
<td>setVelocity</td>
<td>Set velocity</td>
</tr>
<tr>
<td>setWheelPowered</td>
<td>Set wheel powered</td>
</tr>
<tr>
<td>setWheelSpring</td>
<td>Set wheel spring</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>setWheelSteering(int wheel, float steering)</td>
<td>WheeledVehicle</td>
</tr>
<tr>
<td>setWheelTire(int wheel, WheeledVehicleTire tire)</td>
<td>WheeledVehicle</td>
</tr>
<tr>
<td>setWhiteOut(float level)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>skin</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>startFade(int time, int delay, bool fadeOut)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>stopAudio(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>stopThread(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>unmount()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>unmountImage(int slot)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>unmountObject(SceneObject target)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>validateCameraFov(float fov)</td>
<td>ShapeBase</td>
</tr>
<tr>
<td>workingQueryBoxSizeMultiplier</td>
<td>Vehicle</td>
</tr>
<tr>
<td>workingQueryBoxStaleThreshold</td>
<td>Vehicle</td>
</tr>
</tbody>
</table>

Copyright © GarageGames, LLC. All Rights Reserved.
# WheeledVehicleData Member List

This is the complete list of members for `WheeledVehicleData`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom(SimObject fromObject)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>bodyFriction</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>bodyRestitution</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>brakeTorque</code></td>
<td><code>WheeledVehicleData</code></td>
</tr>
<tr>
<td><code>call(string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>cameraDecay</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>cameraDefaultFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraLag</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>cameraMaxDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMaxFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinDist</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraMinFov</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>cameraOffset</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>cameraRoll</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>category</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>checkDeployPos(TransformF txfm)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>collDamageMultiplier</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>collDamageThresholdVel</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>collisionTol</td>
<td>VehicleData</td>
</tr>
<tr>
<td>computeCRC</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>contactTol</td>
<td>VehicleData</td>
</tr>
<tr>
<td>cubeReflectorDesc</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>damageEmitter</td>
<td>VehicleData</td>
</tr>
<tr>
<td>damageEmitterOffset</td>
<td>VehicleData</td>
</tr>
<tr>
<td>damageLevelTolerance</td>
<td>VehicleData</td>
</tr>
<tr>
<td>Debris</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>debrisShapeName</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>density</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>destroyedLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>disabledLevel</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>drag</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dustEmitter</td>
<td>VehicleData</td>
</tr>
<tr>
<td>dustHeight</td>
<td>VehicleData</td>
</tr>
<tr>
<td>engineBrake</td>
<td>WheeledVehicleData</td>
</tr>
<tr>
<td>engineSound</td>
<td>WheeledVehicleData</td>
</tr>
<tr>
<td>engineTorque</td>
<td>WheeledVehicleData</td>
</tr>
<tr>
<td>exitingWater</td>
<td>VehicleData</td>
</tr>
<tr>
<td>exitSplashSoundVelocity</td>
<td>VehicleData</td>
</tr>
<tr>
<td>Explosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>firstPersonOnly</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>Method Name</td>
<td>Class Name</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeployTransform(Point3F pos, Point3F normal)</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hardImpactSound</td>
<td>VehicleData</td>
</tr>
<tr>
<td>hardImpactSpeed</td>
<td>VehicleData</td>
</tr>
<tr>
<td>hardSplashSoundVelocity</td>
<td>VehicleData</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>impactWaterEasy</td>
<td>VehicleData</td>
</tr>
<tr>
<td>impactWaterHard</td>
<td>VehicleData</td>
</tr>
<tr>
<td>impactWaterMedium</td>
<td>VehicleData</td>
</tr>
<tr>
<td>inheritEnergyFromMount</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>integration</td>
<td>VehicleData</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInvincible</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>isMemberOfClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>jetEnergyDrain</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>jetForce</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>jetSound</code></td>
<td><code>WheeledVehicleData</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>mass</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>massBox</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>massCenter</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>maxDamage</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>maxDrag</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>maxEnergy</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>maxSteeringAngle</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>maxWheelSpeed</code></td>
<td><code>WheeledVehicleData</code></td>
</tr>
<tr>
<td><code>mediumSplashSoundVelocity</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>minDrag</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>minImpactSpeed</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>minJetEnergy</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>minRollSpeed</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td>Method</td>
<td>Module</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>numDmgEmitterAreas</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>observeThroughObject</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onAdd(GameBase obj)</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onCollision(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onDamage(ShapeBase obj, float delta)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onDestroyed(ShapeBase obj, string lastState)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onDisabled(ShapeBase obj, string lastState)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onEnabled(ShapeBase obj, string lastState)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onEndSequence(ShapeBase obj, int slot)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onEnterLiquid(Vehicle obj, float coverage, string type)</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>onForceUncloak(ShapeBase obj, string reason)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onImpact(ShapeBase obj, SceneObject collObj, VectorF vec, float len)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onLeaveLiquid(Vehicle obj, string type)</code></td>
<td><code>VehicleData</code></td>
</tr>
<tr>
<td><code>onMount(GameBase obj, SceneObject mountObj, int node)</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onNewDataBlock(GameBase obj)</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onRemove(GameBase obj)</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>onTrigger(ShapeBase obj, int index, bool state)</code></td>
<td><code>ShapeBaseData</code></td>
</tr>
<tr>
<td><code>onUnmount(GameBase obj, SceneObject mountObj, int node)</code></td>
<td><code>GameBaseData</code></td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>reloadOnLocalClient()</code></td>
<td><code>SimDataBlock</code></td>
</tr>
<tr>
<td>Method/Property</td>
<td>Class/Type</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>renderWhenDestroyed</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>repairRate</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td><strong>save</strong>(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>schedule</strong>(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td><strong>setSuperClassNamespace</strong>(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>shadowEnable</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowMaxVisibleDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowProjectionDistance</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSize</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shadowSphereAdjust</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>shapeFile</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>softImpactSound</td>
<td>VehicleData</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>softImpactSpeed</td>
<td>VehicleData</td>
</tr>
<tr>
<td>softSplashSoundVelocity</td>
<td>VehicleData</td>
</tr>
<tr>
<td>splashEmitter</td>
<td>VehicleData</td>
</tr>
<tr>
<td>splashFreqMod</td>
<td>VehicleData</td>
</tr>
<tr>
<td>splashVelEpsilon</td>
<td>VehicleData</td>
</tr>
<tr>
<td>squealSound</td>
<td>WheeledVehicleData</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
<tr>
<td>tireEmitter</td>
<td>WheeledVehicleData</td>
</tr>
<tr>
<td>triggerDustHeight</td>
<td>VehicleData</td>
</tr>
<tr>
<td>underwaterExplosion</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>useEyePoint</td>
<td>ShapeBaseData</td>
</tr>
<tr>
<td>waterWakeSound</td>
<td>VehicleData</td>
</tr>
<tr>
<td>WheelImpactSound</td>
<td>WheeledVehicleData</td>
</tr>
</tbody>
</table>
# WheeledVehicleSpring Member List

This is the complete list of members for `WheeledVehicleSpring`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>antiSwayForce</td>
<td>Method to calculate the anti-sway force.</td>
</tr>
<tr>
<td>assignFieldsFrom</td>
<td>Assign fields from another object.</td>
</tr>
<tr>
<td>assignPersistentId</td>
<td>Assign a persistent ID.</td>
</tr>
<tr>
<td>call</td>
<td>Call a method with arguments.</td>
</tr>
<tr>
<td>canSave</td>
<td>Check if the object can be saved.</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>Check if dynamic fields can be saved.</td>
</tr>
<tr>
<td>class</td>
<td>Get the class name.</td>
</tr>
<tr>
<td>className</td>
<td>Get the class name.</td>
</tr>
<tr>
<td>clone</td>
<td>Create a clone of the object.</td>
</tr>
<tr>
<td>damping</td>
<td>Calculate the damping force.</td>
</tr>
<tr>
<td>deepClone</td>
<td>Deep clone of the object.</td>
</tr>
<tr>
<td>delete</td>
<td>Delete the object.</td>
</tr>
<tr>
<td>dump</td>
<td>Dump the object with detailed information.</td>
</tr>
<tr>
<td>dumpClassHierarchy</td>
<td>Dump the class hierarchy of the object.</td>
</tr>
<tr>
<td>dumpGroupHierarchy</td>
<td>Dump the group hierarchy of the object.</td>
</tr>
<tr>
<td>dumpMethods</td>
<td>Dump all methods of the object.</td>
</tr>
<tr>
<td>force</td>
<td>Calculate the force.</td>
</tr>
<tr>
<td>getCanSave</td>
<td>Get if the object can be saved.</td>
</tr>
<tr>
<td>getClassName</td>
<td>Get the class name.</td>
</tr>
<tr>
<td>getClassNamespace</td>
<td>Get the class namespace.</td>
</tr>
<tr>
<td>getDebugInfo</td>
<td>Get debug information.</td>
</tr>
<tr>
<td>getDeclarationLine</td>
<td>Get the declaration line of the object.</td>
</tr>
<tr>
<td>getDynamicField</td>
<td>Get a dynamic field by index.</td>
</tr>
<tr>
<td>getDynamicFieldCount</td>
<td>Get the count of dynamic fields.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>length</td>
<td>WheeledVehicleSpring</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setHidden(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setLocked(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
</tbody>
</table>
# WheeledVehicleTire Member List

This is the complete list of members for *WheeledVehicleTire*, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assignFieldsFrom</code> <em>(SimObject fromObject)</em></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call</code> <em>(string method, string args...)</em></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deepClone</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dump</code> <em>(bool detailed=false)</em></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getCanSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDeclarationLine</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicField</code> <em>(int index)</em></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDynamicFieldCount</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getField</code> <em>(int index)</em></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldCount</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFieldType</code> <em>(string fieldName)</em></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Function</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td><code>getFieldValue(string fieldName, int index=-1)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getFilename()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getGroup()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getId()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getInternalName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getName()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getSuperClassNamespace()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>hidden</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>internalName</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isChildOfGroup(SimGroup group)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isEditorOnly()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isExpanded()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isField(string fieldName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isInNamespaceHierarchy(string name)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMemberOfClass(string className)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isMethod(string methodName)</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isNameChangeAllowed()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>isSelected()</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>kineticFriction</code></td>
<td><code>WheeledVehicleTire</code></td>
</tr>
<tr>
<td><code>lateralDamping</code></td>
<td><code>WheeledVehicleTire</code></td>
</tr>
<tr>
<td><code>lateralForce</code></td>
<td><code>WheeledVehicleTire</code></td>
</tr>
<tr>
<td><code>lateralRelaxation</code></td>
<td><code>WheeledVehicleTire</code></td>
</tr>
<tr>
<td><code>locked</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>longitudinalDamping</code></td>
<td><code>WheeledVehicleTire</code></td>
</tr>
<tr>
<td><code>longitudinalForce</code></td>
<td><code>WheeledVehicleTire</code></td>
</tr>
<tr>
<td><code>longitudinalRelaxation</code></td>
<td><code>WheeledVehicleTire</code></td>
</tr>
<tr>
<td><code>mass</code></td>
<td><code>WheeledVehicleTire</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method/Property</td>
<td>Class Type</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>radius</td>
<td>WheeledVehicleTire</td>
</tr>
<tr>
<td>reloadOnLocalClient()</td>
<td>SimDataBlock</td>
</tr>
<tr>
<td>restitution</td>
<td>WheeledVehicleTire</td>
</tr>
<tr>
<td>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</td>
<td>SimObject</td>
</tr>
<tr>
<td>schedule(float time, string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setCanSave(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType(string fieldName, string type)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue(string fieldName, string value, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename(string fileName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName(string newInternalName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected(bool state=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setName(string newName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setNameChangeAllowed(bool value=true)</td>
<td>SimObject</td>
</tr>
<tr>
<td>setSuperClassNamespace(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>shapeFile</td>
<td>WheeledVehicleTire</td>
</tr>
<tr>
<td>staticFriction</td>
<td>WheeledVehicleTire</td>
</tr>
<tr>
<td>superClass</td>
<td>SimObject</td>
</tr>
</tbody>
</table>
# ZipObject Member List

This is the complete list of members for `ZipObject`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>addFile</strong>(string filename, string pathInZip, bool replace=true)</td>
<td><code>&lt;object&gt;</code></td>
</tr>
<tr>
<td><code>assignFieldsFrom</code>(SimObject fromObject)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>assignPersistentId</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>call</code>(string method, string args...)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSave</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>canSaveDynamicFields</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>className</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>clone</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>closeArchive</code>()</td>
<td><code>ZipObject</code></td>
</tr>
<tr>
<td><code>closeFile</code>(SimObject stream)</td>
<td><code>ZipObject</code></td>
</tr>
<tr>
<td><code>deepClone</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>delete</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>deleteFile</code>(string pathInZip)</td>
<td><code>ZipObject</code></td>
</tr>
<tr>
<td><code>dump</code>(bool detailed=false)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpClassHierarchy</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpGroupHierarchy</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>dumpMethods</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>extractFile</code>(string pathInZip, string filename)</td>
<td><code>ZipObject</code></td>
</tr>
<tr>
<td><code>getCanSave</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassName</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getClassNamespace</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>getDebugInfo</code>()</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDynamicFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getField(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldCount()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getType(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFieldType(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getValue(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td>getFileEntry(int index)</td>
<td>ZipObject</td>
</tr>
<tr>
<td>getFileEntryCount()</td>
<td>ZipObject</td>
</tr>
<tr>
<td>getFilename()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroup()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getGroupName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getInternalName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getSuperClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method Call</td>
<td>Object</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>openArchive(string filename, string accessMode=&quot;read&quot;)</code></td>
<td>ZipObject</td>
</tr>
<tr>
<td><code>openFileForRead(string filename)</code></td>
<td>ZipObject</td>
</tr>
<tr>
<td><code>openFileForWrite(string filename)</code></td>
<td>ZipObject</td>
</tr>
<tr>
<td><code>parentGroup</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>persistentId</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>save(string fileName, bool selectedOnly=false, string preAppendString=&quot;&quot;)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>schedule(float time, string method, string args...)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setCanSave(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setEditorOnly(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldType(string fieldName, string type)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFieldValue(string fieldName, string value, int index=-1)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setFilename(string fileName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code> setHidden(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setInternalName(string newInternalName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsExpanded(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setIsSelected(bool state=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code> setLocked(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setName(string newName)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setNameChangeAllowed(bool value=true)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>setSuperClassNamespace(string name)</code></td>
<td>SimObject</td>
</tr>
<tr>
<td><code>superClass</code></td>
<td>SimObject</td>
</tr>
</tbody>
</table>
## Zone Member List

This is the complete list of members for Zone, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ambientLightColor</td>
<td>Zone</td>
</tr>
<tr>
<td>assignFieldsFrom(SimObject fromObject)</td>
<td>SimObject</td>
</tr>
<tr>
<td>assignPersistentId()</td>
<td>SimObject</td>
</tr>
<tr>
<td>call(string method, string args...)</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>canSaveDynamicFields</td>
<td>SimObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clearScopeToClient(NetConnection client)</td>
<td>NetObject</td>
</tr>
<tr>
<td>clone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>deepClone()</td>
<td>SimObject</td>
</tr>
<tr>
<td>delete()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dump(bool detailed=false)</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpClassHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpGroupHierarchy()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpMethods()</td>
<td>SimObject</td>
</tr>
<tr>
<td>dumpZoneState(bool updateFirst=true)</td>
<td>Zone</td>
</tr>
<tr>
<td>edge</td>
<td>Zone</td>
</tr>
<tr>
<td>getCanSave()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassName()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClassNamespace()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>getDebugInfo()</td>
<td>SimObject</td>
</tr>
<tr>
<td>getDeclarationLine()</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>getDynamicField</code>(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getDynamicFieldCount</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getEulerRotation</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getField</code>(int index)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldCount</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldType</code>(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFieldValue</code>(string fieldName, int index=-1)</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getFilename</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getForwardVector</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getGhostID</code>()</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getGroup</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getId</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInternalName</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getInverseTransform</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountedObject</code>(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountedObjectCount</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountedObjectNode</code>(int slot)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getMountNodeObject</code>(int node)</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getName</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getObjectBox</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getObjectMount</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getPosition</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getRightVector</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getScale</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td><code>getServerObject</code>()</td>
<td>NetObject</td>
</tr>
<tr>
<td><code>getSuperClassNamespace</code>()</td>
<td>SimObject</td>
</tr>
<tr>
<td><code>getTransform</code>()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>getType()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getUpVector()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBox()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getWorldBoxCenter()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>getZoneId()</td>
<td>Zone</td>
</tr>
<tr>
<td>hidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>internalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>isChildOfGroup(SimGroup group)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isClientObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>isEditorOnly()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isExpanded()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isField(string fieldName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isGlobalBounds()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isInNamespaceHierarchy(string name)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMemberOfClass(string className)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMethod(string methodName)</td>
<td>SimObject</td>
</tr>
<tr>
<td>isMounted()</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isNameChangeAllowed()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isRenderable</td>
<td>Zone</td>
</tr>
<tr>
<td>isRenderEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isSelectable</td>
<td>Zone</td>
</tr>
<tr>
<td>isSelected()</td>
<td>SimObject</td>
</tr>
<tr>
<td>isSelectionEnabled</td>
<td>SceneObject</td>
</tr>
<tr>
<td>isServerObject()</td>
<td>NetObject</td>
</tr>
<tr>
<td>locked</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountNode</td>
<td>SimObject</td>
</tr>
<tr>
<td>mountObject(SceneObject objB, int slot, TransformF txfm=MatrixF::Identity)</td>
<td>SceneObject</td>
</tr>
<tr>
<td>Method/Property</td>
<td>Class</td>
</tr>
<tr>
<td>----------------</td>
<td>-------</td>
</tr>
<tr>
<td>mountPID</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountPos</td>
<td>SceneObject</td>
</tr>
<tr>
<td>mountRot</td>
<td>SceneObject</td>
</tr>
<tr>
<td>name</td>
<td>SimObject</td>
</tr>
<tr>
<td>parentGroup</td>
<td>SimObject</td>
</tr>
<tr>
<td>persistentId</td>
<td>SimObject</td>
</tr>
<tr>
<td>plane</td>
<td>Zone</td>
</tr>
<tr>
<td>point</td>
<td>Zone</td>
</tr>
<tr>
<td>position</td>
<td>SceneObject</td>
</tr>
<tr>
<td>rotation</td>
<td>SceneObject</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>save</td>
<td>SimObject</td>
</tr>
<tr>
<td>scale</td>
<td>SceneObject</td>
</tr>
<tr>
<td>schedule</td>
<td>SimObject</td>
</tr>
<tr>
<td>scopeToClient</td>
<td>NetObject</td>
</tr>
<tr>
<td>setCanSave</td>
<td>SimObject</td>
</tr>
<tr>
<td>setClassNamespace</td>
<td>SimObject</td>
</tr>
<tr>
<td>setEditorOnly</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldType</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFieldValue</td>
<td>SimObject</td>
</tr>
<tr>
<td>setFilename</td>
<td>SimObject</td>
</tr>
<tr>
<td>setHidden</td>
<td>SimObject</td>
</tr>
<tr>
<td>setInternalName</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsExpanded</td>
<td>SimObject</td>
</tr>
<tr>
<td>setIsSelected</td>
<td>SimObject</td>
</tr>
<tr>
<td>setLocked</td>
<td>SimObject</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td><code>setName</code> (string newName)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setNameChangeAllowed</code> (bool value=true)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setScale</code> (Point3F scale)</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>setScopeAlways</code></td>
<td><code>NetObject</code></td>
</tr>
<tr>
<td><code>setSuperClassNamespace</code> (string name)</td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>setTransform</code> (TransformF txfm)</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>soundAmbience</code></td>
<td><code>Zone</code></td>
</tr>
<tr>
<td><code>superClass</code></td>
<td><code>SimObject</code></td>
</tr>
<tr>
<td><code>unmount</code></td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>unmountObject</code> (SceneObject target)</td>
<td><code>SceneObject</code></td>
</tr>
<tr>
<td><code>useAmbientLightColor</code></td>
<td><code>Zone</code></td>
</tr>
<tr>
<td><code>zoneGroup</code></td>
<td><code>Zone</code></td>
</tr>
</tbody>
</table>
- a -

- acceptsAsChild() : SimSet
- activate() : PhysicalZone, SFXState
- activateGhosting() : GameConnection
- activateRow() : GuiGameListMenuCtrl
- activateTurret() : AITurretShape
- add() : BanList, ArrayObject, GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx, SimSet
- addAbsolute() : BanList
- addAutoPlot() : GuiGraphCtrl
- addCategory() : GuiPopUpMenuCtrlEx
- addCollisionDetail() : TSShapeConstructor
- addColumn() : GuiFrameSetCtrl
- addComment() : SimXMLDocument
- addData() : SimXMLDocument
- addDatum() : GuiGraphCtrl
- addFile() : ZipObject
- addFilteredItem() : GuiListBoxCtrl
- addGuiControl() : GuiControl
- addHeader() : SimXMLDocument
- addImposter() : TSShapeConstructor
- addLanguage() : LangTable
- addManager() : RenderPassManager
- addMarker() : SFXSource
- addMenu() : GuiMenuBar
- addMenuItem() : GuiMenuBar
- addMesh() : TSShapeConstructor
- addNewElement() : SimXMLDocument
- addNode() : TSShapeConstructor
- addPage() : GuiTabBookCtrl
- addParameter() : SFXSource
- addPrimitive() : TSShapeConstructor
- addReference() : Message
- addRow() : GuiFrameSetCtrl, GuiGameListMenuCtrl, GuiGameListOptionsCtrl, GuiTextListCtrl
- addScheme() : GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
- addSelection() : GuiTreeViewCtrl
- addSequence() : TSShapeConstructor
- addSubmenuItem() : GuiMenuBar
- addText() : GuiMLTextCtrl, SimXMLDocument
- addTimeOfDayEvent() : TimeOfDay
- addToIgnoreList() : AITurretShape
- addTrigger() : TSShapeConstructor
• allowAllPoses() : Player
• allowCrouching() : Player
• allowJetJumping() : Player
• allowJumping() : Player
• allowProne() : Player
• allowSprinting() : Player
• allowSwimming() : Player
• animate() : TimeOfDay
• animationDone() : PlayerData
• append() : ArrayObject
• apply() : LightDescription, LightFlareData
• applyChanges() : ScatterSky
• applyDamage() : ShapeBase, Lightning
• applyImpulse() : ShapeBase, GameBase
• applyRadialImpulse() : GameBase
• applyRepair() : ShapeBase
• assignFieldsFrom() : SimObject
• assignPersistentId() : SimObject
• attach() : ConsoleLogger, GuiMessageVectorCtrl, GuiWindowCtrl, PhysicsForce
• attachTo() : GuiWindowCtrl
• attachToObject() : ForestWindEmitter
• attribute() : SimXMLDocument
• attributeExists() : SimXMLDocument
• attributeF32() : SimXMLDocument
• attributeS32() : SimXMLDocument
• autoFitRadius() : Camera
- All
- Functions
- Variables

- a
- b
- c
- d
- e
- f
- g
- h
- i
- j
- k
- l
- m
- n
- o
- p
- q
- r
- s
- t
- u
- v
- w
- z

- a -
- aaLevel: RenderFormatToken
- acceleration: SplashData
- accelerator: GuiControl
- accuFire: ShapeBaseImageData
- active: GuiControl, ParticleEmitterNode
- addChildSelectionByValue: GuiTreeViewCtrl
- addressModeU: GFXSamplerStateData
- addressModeV: GFXSamplerStateData
- addressModeW: GFXSamplerStateData
- adjustCenter: TSShapeConstructor
- adjustFloor: TSShapeConstructor
- advancedLightmapSupport: LevelInfo
- airAbsorptionHF: SFXEnvironment
- airControl: PlayerData
- alignDirection: ParticleEmitterData
- alignParticles: ParticleEmitterData
- AlignToTerrain: fxShapeReplicator
- allowColorChars: GuiMLTextCtrl
- allowedMatches: GuiMessageVectorCtrl
- AllowedTerrainSlope: fxFoliageReplicator, fxShapeReplicator
- allowImageStateAnimation: PlayerData
- allowMultipleSelections: GuiListBoxCtrl
- AllowOnInteriors: fxFoliageReplicator, fxShapeReplicator
- AllowOnStatics: fxFoliageReplicator, fxShapeReplicator
- AllowOnTerrain: fxFoliageReplicator, fxShapeReplicator
- AllowOnWater: fxShapeReplicator, fxFoliageReplicator
- allowPlayerStep: fxShapeReplicatedStatic, TSStatic
- allowReflectPass: PostEffect
- allowReorder: GuiTabBookCtrl
- AllowWaterSurface: fxFoliageReplicator, fxShapeReplicator
- alphaArg1: GFXSamplerStateData
- alphaArg2: GFXSamplerStateData
- alphaArg3: GFXSamplerStateData
- AlphaCutoff: fxFoliageReplicator
- alphaDefined: GFXStateBlockData
- alphaOp: GFXSamplerStateData
- alphaRef: Material
- alphaTest: Material
- alphaTestEnable: GFXStateBlockData
- alphaTestFunc: GFXStateBlockData
- alphaTestRef: GFXStateBlockData
- altCommand: GuiControl
- alwaysHandleMouseButtons: GuiCanvas
- alwaysImport: TSShapeConstructor
- alwaysImportMesh: TSShapeConstructor
- ambient: Sun
- ambientFactor: ParticleEmitterData
- ambientLightBlendCurve: LevelInfo
- ambientLightBlendPhase: LevelInfo
- ambientLightColor: Zone
- ambientScale: ScatterSky
- anchorBottom: GuiContainer
- anchorLeft: GuiContainer
- anchorRight: GuiContainer
- anchorTop: GuiContainer
- angularDamping: PhysicsShapeData, PhysicsDebrisData
- angularDrag: Camera, PxMultiActorData
- angularForce: Camera
- angularSleepThreshold: PhysicsShapeData, PhysicsDebrisData
- animate: Sun, LightBase
- animateAllShapes: ShapeBaseImageData
- animateOnServer: ShapeBaseImageData
- animateSplashes: Precipitation
- animateTexture: ParticleData
- animationPeriod: LightBase, LightDescription
- animationPhase: LightBase, LightDescription
- animationType: LightBase, LightDescription
- animFlags: Material
- animSequence: GuiObjectView
- animTexFrames: ParticleData
- animTexName: ParticleData
- animTexTiling: ParticleData
- antiSwayForce: WheeledVehicleSpring
- appliedForce: PhysicalZone
- apply: Sun
- area: MissionArea
- armingDelay: ProjectileData, ProximityMineData
- armingSound : ProximityMineData
- attachments : PxCloth
- attenuationRatio : ScatterSky, Sun, LightDescription, LightBase
- autoAngularForce : FlyingVehicleData
- autoBalance : GuiFrameSetCtrl
- autoCellSize : GuiDynamicCtrlArrayControl
- autoCollapseSiblings : GuiRolloutCtrl
- autoFitExtents : GuiBitmapButtonCtrl
- autoInputDamping : FlyingVehicleData
- autoLinearForce : FlyingVehicleData
- autoSize : GuiIconButtonCtrl
- autoSizeHeight : GuiControlProfile
- autoSizeWidth : GuiControlProfile
- autoSpawn : SpawnSphere
- autoTriggerDelay : ProximityMineData
- axisTilt : TimeOfDay
- azimuth : ScatterSky, Sun
- azimuthOverride : TimeOfDay

Copyright © GarageGames, LLC. All Rights Reserved.
Here is a list of all class members with links to the classes they belong to:
- b -

- backgroundColor : GuiTheoraCtrl
- backSidePassable : Portal
- backwardJetEmitter : FlyingVehicleData
- barBehindText : GuiPaneControl
- baseColor : CloudLayer, WaterObject
- baseRadius : DebrisData
- baseShape : TSShapeConstructor
- baseTex : Material
- baseTexSize : TerrainBlock
- bending : PxCloth
- bendingStiffness : PxCloth
- bevelColorHL : GuiControlProfile
- bevelColorLL : GuiControlProfile
- billboardUVs : GroundCover
- bind() : ActionMap
- bindCmd() : ActionMap
- bindObj() : ActionMap
- binType : RenderBinManager
- bitmap : GuiBitmapCtrl, GuiBitmapButtonCtrl, GuiControlProfile
- GuiTextEditSliderBitmapCtrl, GuiPopUpMenuCtrl,
  GuiPopUpMenuCtrlEx, GuiChunkedBitmapCtrl,
  GuiProgressBitmapCtrl
- bitmapBounds : GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
- bitmapMode : GuiBitmapButtonCtrl
- bitmapName : GuiCursor
- blendDefined : GFXStateBlockData
- blendDest : GFXStateBlockData
- blendEnable : GFXStateBlockData
- blendOp : GFXStateBlockData
- blendSrc : GFXStateBlockData
- blendStyle : ParticleEmitterData
- bodyFriction : RigidShapeData, VehicleData
- bodyRestitution : RigidShapeData, VehicleData
- boltStartRadius : Lightning
- border : GuiControlProfile
- borderColor : GuiControlProfile, GuiFrameSetCtrl
- borderColorHL : GuiControlProfile
- borderColorNA: GuiControlProfile
- borderEnable: GuiFrameSetCtrl
- borderMargin: GuiSeparatorCtrl
- borderMovable: GuiFrameSetCtrl
- borderThickness: GuiControlProfile
- borderWidth: GuiFrameSetCtrl
- bottomMaterial: MeshRoad
- bounceElasticity: ProjectileData
- bounceFriction: ProjectileData
- bounceVariance: DebrisData
- boundingBox: GameBase, PlayerData
- boxHeadBackPercentage: PlayerData
- boxHeadFrontPercentage: PlayerData
- boxHeadLeftPercentage: PlayerData
- boxHeadPercentage: PlayerData
- boxHeadRightPercentage: PlayerData
- boxHeight: Precipitation
- boxTorsoPercentage: PlayerData
- boxWidth: Precipitation
- brakeMultiplier: Camera
- brakeTorque: WheeledVehicleData
- brakingActivationSpeed: HoverVehicleData
- brakingForce: HoverVehicleData
- branchAmp: ForestItemData
- breakAngle: MeshRoad, DecalRoad
- breakForce: PxMultiActorData
- brightness: ScatterSky, LightBase, Sun, LightDescription
- brightnessA: LightAnimData
- brightnessKeys: LightAnimData
- brightnessPeriod: LightAnimData
- brightnessSmooth: LightAnimData
- brightnessZ: LightAnimData
- bringToFront(): SimSet
- broken: PxMultiActor
- bubbleEmitTime: PlayerData
- buildIconTable(): GuiTreeViewCtrl
- buildVisibleTree: GuiTreeViewCtrl
- bumpAtlas: Material
- bumpTex: Material
- buoyancyDensity: PxMultiActorData, PhysicsDebrisData, PhysicsShapeData
- buttonMargin: GuiIconButtonCtrl
- buttonType: GuiButtonBaseCtrl

Copyright © GarageGames, LLC. All Rights Reserved.
Here is a list of all class members with links to the classes they belong to:
- c -

- calculateViewDistance() : GuiTSCtrl
- call() : SimObject
- callbackOnA : GuiGameListMenuCtrl
- callbackOnB : GuiGameListMenuCtrl
- callbackOnX : GuiGameListMenuCtrl
- callbackOnY : GuiGameListMenuCtrl
- callOnChildren() : SimSet
- callOnChildrenNoRecurse() : SimSet
- cameraDecay : RigidShapeData, VehicleData
- cameraDefaultFov : ShapeBaseData
- cameraLag : RigidShapeData, VehicleData
- cameraMaxDist : ShapeBaseData
- cameraMaxFov : ShapeBaseData
- cameraMinDist : ShapeBaseData
- cameraMinFov : ShapeBaseData
- cameraOffset : RigidShapeData, TurretShapeData, VehicleData
- cameraRoll : RigidShapeData, VehicleData
- cameraSpeed : GuiObjectView
- cameraZRot : GuiTSCtrl
- camShakeAmp : ExplosionData, ShapeBaseImageData
- camShakeDuration : ExplosionData
- camShakeFalloff : ExplosionData
- camShakeFreq : ExplosionData, ShapeBaseImageData
- camShakeRadius : ExplosionData
- cancelRename : GuiTreeViewCtrl
- canCloak() : ShapeBase
- canClose : GuiWindowCtrl
- canCollapse : GuiWindowCtrl
- canKeyFocus : GuiControlProfile
- canMaximize : GuiWindowCtrl
- canMinimize : GuiWindowCtrl
- canMove : GuiWindowCtrl
- canRenameObject() : GuiTreeViewCtrl
- canRenameObjects : GuiTreeViewCtrl
- canSave : SimObject
- canSaveDynamicFields : SimObject
- canvasClearColor: LevelInfo
- caption: GuiSeparatorCtrl, GuiRolloutCtrl, GuiPaneControl
- captionID: GuiPaneControl
- caseSensitive: ArrayObject
- casing: ShapeBaseImageData
- castShadows: LightBase, Material, LightDescription, PhysicsDebrisData, ScatterSky, Sun, TerrainBlock
- category: GameBaseData, GuiControlProfile
- cellIndex: Material
- cellLayout: Material
- cellSize: Material
- center: GuiSpeedometerHud
- centerY: GuiGraphCtrl
- chanceToHitTarget: Lightning
- changeChildPosition: GuiStackControl
- changeChildSizeToFit: GuiStackControl
- changeMaterial(): ShapeBase, TSSstatic, InteriorInstance
- changePath: FileDialog
- changeTextById(): GuiPopUpMenuCtrl
- channel: SFXParameter
- chaseCam(): GameConnection
- checkDeployPos(): ShapeBaseData
- checkDismountPoint(): Player
- checkMaxRate(): NetConnection
- childBorder: GuiAutoScrollCtrl
- childMargin: GuiScrollCtrl
- clarity: WaterObject
- class: SimObject
- className: SimObject
- clear: GuiPopUpMenuCtrl, SimSet, SimXMLDocument, Forest, GuiTreeViewCtrl, GuiPopUpMenuCtrlEx, GuiTextListCtrl, MessageVector, PfxVis
- clearAim(): AlPlayer
- clearAllOnSingleSelection: GuiTreeViewCtrl
- clearCameraObject(): GameConnection
- clearControlObject(): Player
- clearEntry(): GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
- clearError(): SimXMLDocument
- clearFilterText(): GuiTreeViewCtrl
- clearFirstResponder() : GuiControl
- clearItemColor() : GuiListBoxCtrl
- clearItems() : GuiListBoxCtrl
- clearMenuItems() : GuiMenuBar
- clearMenus() : GuiMenuBar
- clearPaths() : NetConnection
- clearScopeToClient() : NetObject
- clearSelectedText() : GuiTextEditCtrl
- clearSelection() : GuiListBoxCtrl, GuiTreeCtrl, GuiTextCtrl
- clearShaderMacros() : PostEffect
- clearSubmenuItems() : GuiMenuBar
- click() : GuiFadeinBitmapCtrl
- clickCollapse : GuiRolloutCtrl
- clientOnly : PxMultiActorData
- clientSide : TriggerData
- clientToScreen() : GuiCanvas
- clipColumnText : GuiTextListCtrl
- clippingAngle : DecalData
- cloakable : ShapeBaseImageData
- clone() : SimObject
- close() : FileStreamObject, FileObject
- closeArchive() : ZipObject
- closeCommand : GuiWindowCtrl
- closeFile() : ZipObject
- clumpExponent : GroundCover
- clumpRadius : GroundCover
- colCount : GuiControlArrayControl, GuiDynamicCtrlArrayControl
- collapsable : GuiPaneCtrl
- collapse() : GuiRolloutCtrl
- collDamageMultiplier : VehicleData
- collDamageThresholdVel : VehicleData
- collidable : ForestItemData
- collisionTol : RigidShapeData, VehicleData
- collisionType : fxShapeReplicatedStatic, TSStatic
- color : LightBase, GuiSpeedometerHud, LightDescription, Sun, GuiSwatchButtonCtrl, Lightning
- colorA : LightAnimData
- colorArg1: GFXSamplerStateData
- colorArg2: GFXSamplerStateData
- colorArg3: GFXSamplerStateData
- colorBullet: GuiListBoxCtrl
- colorKeys: LightAnimData
- colorMultiply: Material
- colorPeriod: LightAnimData
- colors: ParticleData, SplashData
- colorSmooth: LightAnimData
- colorWriteAlpha: GFXStateBlockData
- colorWriteBlue: GFXStateBlockData
- colorWriteDefined: GFXStateBlockData
- colorWriteGreen: GFXStateBlockData
- colorWriteRed: GFXStateBlockData
- colorZ: LightAnimData
- colSize: GuiDynamicCtrlArrayControl
- colSizes: GuiControlArrayControl
- colSpacing: GuiDynamicCtrlArrayControl, GuiControlArrayControl
- columns: GuiTextListCtrl, GuiFrameSetCtrl
- columnSplit: GuiGameListOptionsProfile
- command: GuiControl
- compareToObjectID: GuiTreeViewCtrl
- computeCRC: ShapeBaseData, ShapeBaseImageData
- computeSizes(): GuiScrollCtrl
- coneInsideAngle: SFXEmitter, SFXDescription
- coneOutsideAngle: SFXDescription, SFXEmitter
- coneOutsideVolume: SFXEmitter, SFXDescription
- connect(): NetConnection, TCPObj
- connectLocal(): NetConnection
- constantAcceleration: ParticleData
- constantThumbHeight: GuiScrollCtrl
- contactTol: RigidShapeData, VehicleData
- controlsIsChild(): GuiControl
- controlMode: Camera
- cookie: LightBase, Sun, LightDescription, ScatterSky
- copyEffect: RenderFormatToken
- copyFrom(): StreamObject
- coronaEnabled: Sun
• coronaMaterial : Sun
• coronaScale : Sun
• coronaTint : Sun
• coronaUseLightColor : Sun
• correctMuzzleVector : ShapeBaseImageData
• correctMuzzleVectorTP : ShapeBaseImageData
• count() : ArrayObject
• countKey() : ArrayObject
• countValue() : ArrayObject
• coverage : CloudLayer
• createHoverHeight : FlyingVehicleData
• createNew : TerrainBlock
• createNullDevice() : GFXInit
• crop() : ArrayObject
• crouchBoundingBox : PlayerData
• crouchForce : PlayerData
• crouchTrigger : Player
• cubeFace : CubemapData
• cubemap : Material, WaterObject
• cubeReflectorDesc : ShapeBaseData
• cullDefined : GFXStateBlockData
• cullMode : GFXStateBlockData
• CullResolution : fxFoliageReplicator
• cursorColor : GuiControlProfile
• cursorOff() : GuiCanvas
• cursorOn() : GuiCanvas
• customFootstepSound : Material
• customImpactSound : Material

Copyright © GarageGames, LLC. All Rights Reserved.
Here is a list of all class members with links to the classes they belong to:
- d -

- damageEmitter : VehicleData
- damageEmitterOffset : VehicleData
- damageFillColor : GuiHealthBarHud, GuiCrossHairHud
- damageFrameColor : GuiCrossHairHud
- damageLevelTolerance : VehicleData
- damageOffset : GuiCrossHairHud
- damageRect : GuiCrossHairHud
- damping : PxCloth, WheeledVehicleSpring
- dampingCoefficient : PxCloth, ForestItemData
- dataBlock : GameBase
- dataFile : Forest
- dayLength : TimeOfDay
- dayScale : TimeOfDay
- deactivate() : PhysicalZone, SFXState
- deactivateTurret() :AITurretShape
- Debris : ShapeBaseData, ExplosionData, PhysicsShapeData
- debrisNum : ExplosionData
- debrisNumVariance : ExplosionData
- debrisPhiMax : ExplosionData
- debrisPhiMin : ExplosionData
- debrisShapeName : ShapeBaseData
- debrisThetaMax : ExplosionData
- debrisThetaMin : ExplosionData
- debrisVelocity : ExplosionData
- debrisVelocityVariance : ExplosionData
- DebugBoxHeight : fxFoliageReplicator
- debugRender : RenderOcclusionMgr, PxMultiActor, GuiGameListMenuCtrl, TerrainBlock
- decal : ProjectileData
- decalBias : LevelInfo
- DecalData : PlayerData
- decalOffset : PlayerData
- decalType : fxShapeReplicatedStatic, TSSStatic
- decayHFRatio : SFXEnvironment
- decayLFRatio : SFXEnvironment
- decayTime : SFXEnvironment
- deepClone() : SimObject
- defaultFile: `FileDialog`
- defaultHeight: `Gui rollout Ctrl`
- defaultPage: `Gui tab book Ctrl`
- defaultPath: `FileDialog`
- defaultValue: `SFX Parameter`
- defines: `ShaderData`
- delayMS: `ExplosionData, SplashData`
- delayTimeIn: `SFX PlayList`
- delayTimeInVariance: `SFX PlayList`
- delayTimeOut: `SFX PlayList`
- delayTimeOutVariance: `SFX PlayList`
- delayVariance: `ExplosionData, SplashData`
- delete(): `Sim Object, Game Connection`
- deleteAllObjects(): `Sim Set`
- deleteFile(): `Zip Object`
- deleteItem(): `Gui ListBox Ctrl`
- deleteLine(): `Message Vector`
- deleteObjectAllowed: `Gui Tree View Ctrl`
- deleteOnMouseUp: `Gui Drag And Drop Control`
- deleteSelection(): `Gui Tree View Ctrl`
- deniedSound: `Gui Text Edit Ctrl, Gui ML Text Ctrl`
- density: `Shape Base Data, Px Cloth, Water Object, SFX Environment`
- depthFormat: `Render Format Token`
- depthGradientMax: `Water Object`
- depthGradientTex: `Water Object`
- description: `SFX Track, SFX Source, SFX Parameter`
- destroy(): `Physics Shape`
- destroyedLevel: `Shape Base Data`
- destroyedShape: `Physics Shape Data`
- destroyThread(): `Shape Base`
- destroyTreeOnSleep: `Gui Tree View Ctrl`
- detach(): `Physics Force, Console Logger, Gui Message Vector Ctrl`
- detailAdjust: `Reflector Desc`
- detailAmp: `Forest Item Data`
- detailDistance: `Terrain Material`
- detailFreq: `Forest Item Data`
- detailMap: `Terrain Material, Material`
- detailNormalMap : Material
- detailNormalMapStrength : Material
- detailScale : Material
- detailSize : TerrainMaterial
- detailStrength : TerrainMaterial
- detailTex : Material
- diffuseColor : Material
- diffuseMap : Material, TerrainMaterial
- diffuseSize : TerrainMaterial
- diffusion : SFXEnvironment
- disable() : RenderPassStateToken, PostEffect, SFXState
- disabledLevel : ShapeBaseData
- disableImposters : Forest
- disableMove : Vehicle
- discardAll : DecalRoad
- disconnect() : TCPObject
- displayEnergy : GuiHealthBarHud
- dissolveRadius : GroundCover
- distanceFade : GuiShapeNameHud
- distortEndDist : WaterObject
- distortFullDepth : WaterObject
- distortStartDist : WaterObject
- docking : GuiContainer
- doCollision : Precipitation
- doDismount() : PlayerData
- doMirror() : GuiListBoxCtrl
- done : GuiFadeinBitmapCtrl
- dopplerFactor : SFXAmbience
- doRespawn() : TurretShape
- doubleSided : Material
- downJetEmitter : FlyingVehicleData
- drag : ShapeBaseData, Camera
- dragCoefficient : ParticleData
- dragForce : RigidShapeData, HoverVehicleData
- dragToItemAllowed : GuiTreeViewCtrl
- drawBottom : SkyBox
- drawBounds : Forest
- drawBox() : DebugDrawer
- drawCells : Forest
- `drawLine()`: DebugDrawer
- `dropAnimateMS`: Precipitation
- `dropShader`: PrecipitationData
- `dropSize`: Precipitation
- `dropsPerSide`: PrecipitationData
- `dropTexture`: PrecipitationData
- `dump()`: SimObject, MessageVector
- `dumpClassHierarchy()`: SimObject
- `dumpEvents()`: EventManager
- `dumpGroupHierarchy()`: SimObject
- `dumpInstances`: Material
- `dumpMeshVisibility()`: ShapeBase
- `dumpMethods()`: SimObject
- `dumpModel`: PxMultiActorData
- `dumpShaderDisassembly()`: PostEffect
- `dumpShape()`: TSShapeConstructor
- `dumpSubscribers()`: EventManager
- `dumpZoneState()`: Zone
- `duplicate()`: ArrayObject
- `dustEmitter`: RigidShapeData, PlayerData, VehicleData
- `dustHeight`: RigidShapeData, VehicleData
- `dustTrailEmitter`: RigidShapeData, HoverVehicleData
- `dustTrailFreqMod`: HoverVehicleData
- `dustTrailOffset`: HoverVehicleData
- `DXPixelShaderFile`: ShaderData
- `DXVertexShaderFile`: ShaderData
- `dynamic`: CubemapData
- `dynamicCubemap`: Material
- `dynamicFarDist`: CubemapData
- `dynamicFriction`: PxMaterial
- `dynamicNearDist`: CubemapData
- `dynamicNonStackExtent`: GuiStackControl
- `dynamicObjectTypeMask`: CubemapData
- `dynamicPos`: GuiStackControl
- `dynamicSize`: GuiDynamicCtrlArrayControl, CubemapData, GuiStackControl
- `dynamicType`: StaticShapeData
Here is a list of all class members with links to the classes they belong to:
- e -

- echo() : ArrayObject
- echoDepth : SFXEnvironment
- echoTime : SFXEnvironment
- edge : OcclusionVolume, SFXSpace, Zone
- edgeSnap : GuiWindowCtrl
- editItem() : GuiTreeViewCtrl
- EditorOpen : DecalRoad, MeshRoad, River
- effectColor : Material
- ejectionAngle : SplashData
- ejectionFreq : SplashData
- ejectionOffset : ParticleEmitterData
- ejectionPeriodMS : ParticleEmitterData
- ejectionVelocity : ParticleEmitterData
- elasticity : DebrisData, ItemData
- elementDist : LightFlareData
- elementRect : LightFlareData
- elementRotate : LightFlareData
- elementScale : LightFlareData
- elementTint : LightFlareData
- elementUseLightColor : LightFlareData
- elementValue() : SimXMLDocument
- elevation : ScatterSky, Sun
- elevationMax : ForestBrushElement
- elevationMin : ForestBrushElement
- emap : ShapeBaselImageData
- emissive : WaterObject, Material
- emitter : ExplosionData, SplashData, ParticleEmitterNode
- emitters : DebrisData
- empty() : ArrayObject
- enable() : RenderPassStateToken, PostEffect, SFXState
- enabled : RenderPassStateToken
- engineBrake : WheeledVehicleData
- engineSound : FlyingVehicleData, HoverVehicleData, WheeledVehicleData
- engineTorque : WheeledVehicleData
- enterCommand : Trigger
- envDiffusion : SFXEnvironment
- environment : SFXAmbience
- envMap : Material
- envSize : SFXEnvironment
- envTex : Material
- erase() : ArrayObject
- escapeCommand : GuiTextEditCtrl, GuiMLTextEditCtrl
- excludedStates : SFXState
- Execute() : FileDialog
- exitingWater : PlayerData, RigidShapeData, VehicleData
- exitSplashSoundVelocity : PlayerData, RigidShapeData, VehicleData
- expand() : GuiRolloutCtrl
- expanded : GuiRolloutCtrl
- expandItem() : GuiTreeViewCtrl
- explode() : ProximityMine
- explodeOnMaxBounce : DebrisData
- Explosion : SplashData, PhysicsShapeData, ProjectileData, ShapeBaseData, DebrisData
- explosionOffset : ProximityMineData
- explosionScale : ExplosionData
- explosionShape : ExplosionData
- export() : BanList
- exportHeightMap() : TerrainBlock
- exportLayerMaps() : TerrainBlock
- exportToCollada() : InteriorInstance
- exposure : CloudLayer, ScatterSky
- extent : GuiControl
- extractFile() : ZipObject
- eyeOffset : ShapeBaseImageData
- eyeRotation : ShapeBaseImageData
Here is a list of all class members with links to the classes they belong to:
- f -

- faceViewer : ExplosionData
- fade : DebrisData
- fadeColor : GuiFadeinBitmapCtrl, Lightning
- fadeDelay : ProjectileData
- fadeDist : Precipitation
- fadeDistEnd : Precipitation
- fadeEndPixelSize : DecalData
- fadeInEase : SFXDescription, GuiFadeinBitmapCtrl
- FadeInRegion : fxFoliageReplicator
- fadeInTime : SFXDescription, SFXEmitter, GuiFadeinBitmapCtrl
- fadeLoops : SFXDescription
- fadeOutEase : SFXDescription, GuiFadeinBitmapCtrl
- FadeOutRegion : fxFoliageReplicator
- fadeOutTime : SFXDescription, SFXEmitter, GuiFadeinBitmapCtrl
- fadeStartDistance : LightBase, LightDescription, ScatterSky, Sun
- fadeStartPixelSize : DecalData
- fadeTime : DecalData
- fadeTimeIn : SFXPlayList
- fadeTimeInVariance : SFXPlayList
- fadeTimeOut : SFXPlayList
- fadeTimeOutVariance : SFXPlayList
- fallback : CustomMaterial
- fallingSpeedThreshold : PlayerData
- farDist : ReflectorDesc
- ffLighting : GFXStateBlockData
- fileFilter : GuiDirectoryFileListCtrl
- fileMustExist : OpenFolderDialog
- fileName : OpenFileDialog, SFXFMODProject, Prefab, SFXEmitter, SFXProfile
- filePath : GuiDirectoryFileListCtrl
- fillColor : GuiHealthBarHud, GuiHealthTextHud, GuiShapeNameHud, GuiControlProfile, GuiClockHud
- fillColorHL : GuiControlProfile
- fillColorNA : GuiControlProfile
- fillColorSEL: GuiControlProfile
- fillRowFirst: GuiDynamicCtrlArrayControl
- filters: FileDialog
- findChildItemByName(): GuiTreeViewCtrl
- findHitControl(): GuiControl
- findHitControls(): GuiControl
- findItemByName(): GuiTreeViewCtrl
- findItemByObjectId(): GuiTreeViewCtrl
- findItemByValue(): GuiTreeViewCtrl
- findItemText(): GuiListBoxCtrl
- findObjectByInternalName(): SimSet
- findText(): GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
- findTextIndex(): GuiTextListCtrl
- firstAttribute(): SimXMLDocument
- firstPerson: ShapeBaseImageData
- firstPersonOnly: ShapeBaseData
- firstPersonShadows: PlayerData
- fitBook: GuiTabPageCtrl
- fitParentWidth: GuiListBoxCtrl, GuiTextListCtrl
- FixAspectRatio: fxFoliageReplicator
- fixedPanel: GuiSplitContainer
- fixedSize: GuiSplitContainer
- FixSizeToMax: fxFoliageReplicator
- flags: SFXEnvironment
- flareEnabled: LightFlareData
- flareScale: LightBase, LightDescription, ScatterSky, Sun
- flareTexture: LightFlareData
- flareType: LightBase, LightDescription, ScatterSky, Sun
- flightCeiling: MissionArea
- flightCeilingRange: MissionArea
- floatingGravMag: HoverVehicleData
- floatingThrustFactor: HoverVehicleData
- floatSound: HoverVehicleData
- FlowMagnitude: River
- flush: Material
- fmodGroup: SFXFMODEvent, SFXFMODEventGroup
- fmodName: SFXFMODEvent, SFXFMODEventGroup
- fmodParameterRanges: SFXFMODEvent
- fmodParameterValues: SFXFMODEvent
- fmodProject: SFXFMODEventGroup
- foamAmbientLerp: WaterObject
- foamDir: WaterObject
- foamMaxDepth: WaterObject
- foamOpacity: WaterObject
- foamRippleInfluence: WaterObject
- foamSpeed: WaterObject
- foamTex: WaterObject
- foamTexScale: WaterObject
- focusOnMouseWheel: GuiTextEditSliderBitmapCtrl, GuiTextEditSliderCtrl
- fogAtmosphereHeight: LevelInfo
- fogBandHeight: SkyBox
- fogColor: LevelInfo
- fogDensity: LevelInfo
- fogDensityOffset: LevelInfo
- fogScale: ScatterSky
- FoliageCount: fxFoliageReplicator
- FoliageFile: fxFoliageReplicator
- FoliageRetries: fxFoliageReplicator
- followCam: Precipitation
- fontCharset: GuiControlProfile
- fontColor: GuiControlProfile
- fontColorHL: GuiControlProfile
- fontColorLink: GuiControlProfile
- fontColorLinkHL: GuiControlProfile
- fontColorNA: GuiControlProfile
- fontColors: GuiControlProfile
- fontColorSEL: GuiControlProfile
- fontSize: GuiControlProfile
- fontType: GuiControlProfile
- FootBubblesSound: PlayerData
- FootHardSound: PlayerData
- FootMetalSound: PlayerData
- footPuffEmitter: PlayerData
- footPuffNumParts: PlayerData
- footPuffRadius: PlayerData
- FootShallowSound: PlayerData
- FootSnowSound: PlayerData
- FootSoftSound: PlayerData
- footstepSoundId: Material
- footstepSplashHeight: PlayerData
- FootUnderwaterSound: PlayerData
- FootWadingSound: PlayerData
- force: WheeledVehicleSpring, Camera
- forceClose: GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
- forceDetail: fxShapeReplicatedStatic, TSStatic
- forceFOV: GuiTSCtrl
- forceImposters: Forest
- forceOnAction: GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
- forceReflow(): GuiMLTextCtrl
- forceUpdateMaterials: TSShapeConstructor
- forceValidateText(): GuiTextEditCtrl
- ForestItemData: ForestBrushElement
- format: RenderFormatToken, GuiTextEditSliderBitmapCtrl, GuiTextEditSliderCtrl
- forwardJetEmitter: HoverVehicleData, FlyingVehicleData
- forwardLit: CustomMaterial
- frame: DecalData
- frameBorder(): GuiFrameSetCtrl
- frameColor: GuiShapeNameHud, GuiHealthBarHud, GuiHealthTextHud, GuiClockHud
- frameMinExtent(): GuiFrameSetCtrl
- frameMovable(): GuiFrameSetCtrl
- framePadding(): GuiFrameSetCtrl
- framesPerSec: ParticleData
- freeData(): SFXFMODEventGroup
- freeReference(): Message
- freeze(): GuiStackControl
- freezeSim(): RigidShape
- fresnelBias: WaterObject
- fresnelPower: WaterObject
- friction: PhysicsDebrisData, PhysicsShapeData, DebrisData, ItemData, PxCloth
- frontSidePassable: Portal
- frontTabPadding: GuiTabBookCtrl
- frozen: GuiDynamicCtrlArrayControl
- fudgeFactor: GuiFrameSetCtrl
• fullReflect : WaterObject
• fullRowSelect : GuiTreeViewCtrl

Copyright © GarageGames, LLC. All Rights Reserved.
Here is a list of all class members with links to the classes they belong to:
- **g** -

- `get()`: `HTTPObject`
- `getAdapterCount()`: `GFXInit`
- `getAdapterMode()`: `GFXInit`
- `getAdapterModeCount()`: `GFXInit`
- `getAdapterName()`: `GFXInit`
- `getAdapterShaderModel()`: `GFXInit`
- `getAdapterType()`: `GFXInit`
- `getAddress`: `AIConnection`, `NetConnection`
- `getAimLocation`: `AIClient`, `AIPlayer`
- `getAimObject()`: `AIPlayer`
- `getAIRepairPoint()`: `ShapeBase`
- `getAllowManualFire()`: `TurretShape`
- `getAllowManualRotation()`: `TurretShape`
- `getAngularVelocity()`: `Camera`
- `getAnimFlags`: `Material`
- `getArea()`: `MissionArea`
- `getAspect()`: `GuiControl`
- `getAspectRatio()`: `PostEffect`
- `getAttenuatedVolume()`: `SFXSource`
- `getBinding()`: `ActionMap`
- `getBinType()`: `RenderBinManager`
- `getBounds()`: `TShapeConstructor`
- `getCameraFov()`: `ShapeBase`
- `getCameraObject()`: `GameConnection`
- `getCameraSpeed()`: `GuiObjectView`
- `getCanSave()`: `SimObject`
- `getCard()`: `GFXCardProfilerAPI`
- `getCenter()`: `GuiControl`
- `getChild()`: `GuiTreeviewCtrl`
- `getClassName()`: `SimObject`
- `getClassNamespace()`: `SimObject`
- `getClientObject()`: `NetObject`
- `getColorById()`: `GuiPopUpMenuCtrlEx`
- `getColumnCount()`: `GuiFrameSetCtrl`
- `getColumnOffset()`: `GuiFrameSetCtrl`
- `getCommand()`: `ActionMap`
- `getContent()`: `GuiCanvas`
- `getControlCameraDefaultFov() : GameConnection`
- `getControlCameraFov() : GameConnection`
- `getControllingClient() : ShapeBase`
- `getControllingObject() : ShapeBase`
- `getControlObject() : GameConnection, Player`
- `getCount() : SimSet`
- `getCurrent() : ArrayObject`
- `getCurrentLanguage() : LangTable`
- `getCurrentOption() : GuiGameListOptionsCtrl`
- `getCurrentSlot() : SFXController`
- `getCurrentTime() : GuiTheoraCtrl`
- `getCursorPos() : GuiTextEditCtrl, GuiCanvas`
- `getDamageFlash() : ShapeBase, GameConnection`
- `getDamageLevel() : ShapeBase`
- `getDamageLocation() : Player`
- `getDamagePercent() : ShapeBase`
- `getDamageState() : ShapeBase`
- `getData() : SimXMLDocument`
- `getDataBlock() : GameBase`
- `getDatum() : GuiGraphCtrl`
- `getDeadZone() : ActionMap`
- `getDebugInfo() : SimObject`
- `getDeclarationLine() : SimObject`
- `getDefaultAdapterIndex() : GFXInit`
- `getDefaultCameraFov() : ShapeBase`
- `getDeployTransform() : ShapeBaseData`
- `getDetailLevelCount() : TSShapeConstructor`
- `getDetailLevelIndex() : TSShapeConstructor`
- `getDetailLevelName() : TSShapeConstructor`
- `getDetailLevelSize() : TSShapeConstructor`
- `getDuration() : SFXSound`
- `getDynamicField() : SimObject`
- `getDynamicFieldCount() : SimObject`
- `getEnergyLevel() : ShapeBase`
- `getEnergyPercent() : ShapeBase`
- `getErrorDesc() : SimXMLDocument`
- `getEulerRotation() : SceneObject`
- `getExtent() : GuiControl, GuiCanvas`
- `getEyePoint() : ShapeBase`
- `getEyeTransform()`: `ShapeBase`
- `getEyeVector()`: `ShapeBase`
- `getFadeInTime()`: `SFXSource`
- `getFadeOutTime()`: `SFXSource`
- `getField()`: `SimObject`
- `getFieldCount()`: `SimObject`
- `getFieldType()`: `SimObject`
- `getFieldValue()`: `SimObject`
- `getFileEntry()`: `ZipObject`
- `getFileEntryCount()`: `ZipObject`
- `getFilename`: `Material`, `SimObject`, `CubemapData`
- `getFilterText()`: `GuiTreeCtrl`
- `getFirstResponder()`: `GuiControl`
- `getFirstRootItem`: `GuiTreeCtrl`
- `getForwardVector()`: `SceneObject`
- `getFramePadding()`: `GuiFrameSetCtrl`
- `getFreeLook`: `AIConnection`
- `getFullCount()`: `SimSet`
- `getGhostID()`: `NetObject`, `NetConnection`
- `getGhostsActive()`: `NetConnection`
- `getGlobalCenter()`: `GuiControl`
- `getGlobalPosition()`: `GuiControl`
- `getGroup()`: `SimObject`
- `getId()`: `SimObject`
- `getImageAltTrigger()`: `ShapeBase`
- `getImageAmmo()`: `ShapeBase`
- `getImageGenericTrigger()`: `ShapeBase`
- `getImageLoaded()`: `ShapeBase`
- `getImageScriptAnimPrefix()`: `ShapeBase`
- `getImageSkinTag()`: `ShapeBase`
- `getImageState()`: `ShapeBase`
- `getImageTarget()`: `ShapeBase`
- `getImageTrigger()`: `ShapeBase`
- `getImposterDetailLevel()`: `TSShapeConstructor`
- `getImposterSettings()`: `TSShapeConstructor`
- `getIndexFromKey()`: `ArrayObject`
- `getIndexFromValue()`: `ArrayObject`
- `getInternalName()`: `SimObject`
- `getInverseTransform()`: `SceneObject`
- getItemCount : GuiTreeCtrl, GuiListCtrl
- getItemObject() : GuiListCtrl
- getItemText() : GuiListCtrl, GuiTreeCtrl
- getItemValue() : GuiTreeCtrl
- getKey() : ArrayObject
- getLangName() : LangTable
- getLastClickItem() : GuiListCtrl
- getLastStickyNormal() : Item
- getLastStickyPos() : Item
- getLineIndexByTag() : MessageVector
- getLineTag() : MessageVector
- getLineText() : MessageVector
- getLineTextByTag() : MessageVector
- getLocation : AIClient
- getLookAtPoint() : ShapeBase
- getManager() : RenderPassManager
- getManagerCount() : RenderPassManager
- getMaxDamage() : ShapeBase
- getMeshCount() : TSShapeConstructor
- getMeshMaterial() : TSShapeConstructor
- getMeshName() : TSShapeConstructor
- getMeshSize() : TSShapeConstructor
- getMeshType() : TSShapeConstructor
- getMinExtent() : GuiControl
- getMode() : Camera, GuiCanvas
- getModeCount() : GuiCanvas
- getModel() : GuiObjectView
- getModelFile() : ShapeBase, TSStatic, InteriorInstance
- getMountedImage() : ShapeBase
- getMountedModel() : GuiObjectView
- getMountedObject() : SceneObject
- getMountedObjectCount() : SceneObject
- getMountedObjectName() : SceneObject
- getMountNodeObject() : SceneObject
- getMountSkin() : GuiObjectView
- getMountSlot() : ShapeBase
- getMouseControl() : GuiCanvas
- getMove() : AIConnection
- getMoveDestination : AIClient, AIPlayer
- `getMoveSpeed()` : `AIPlayer`
- `getMuzzlePoint()` : `ShapeBase`
- `getMuzzleVector()` : `ShapeBase`
- `getName()` : `SimObject`
- `getNextSibling()` : `GuiTreeViewCtrl`
- `getNodeChildCount()` : `TSShapeConstructor`
- `getNodeChildName()` : `TSShapeConstructor`
- `getNodeCount()` : `TSShapeConstructor`
- `getNodeIndex()` : `TSShapeConstructor`
- `getNodeName()` : `TSShapeConstructor`
- `getNodeObjectCount()` : `TSShapeConstructor`
- `getNodeObjectName()` : `TSShapeConstructor`
- `getNodeParentName()` : `TSShapeConstructor`
- `getNodeTransform()` : `TSShapeConstructor`
- `getNumDeathAnimations()` : `Player`
- `getNumDetailLevels()` : `InteriorInstance`
- `getNumLang()` : `LangTable`
- `getNumLines()` : `MessageVector`
- `getNumObjects()` : `Trigger`
- `getObject()` : `SimSet, Trigger`
- `getObjectBox()` : `SceneObject`
- `getObjectCount()` : `TSShapeConstructor`
- `getObjectIndex()` : `SimSet, TSShapeConstructor`
- `getObjectMount()` : `SceneObject`
- `getObjectName()` : `TSShapeConstructor`
- `getObjectNode()` : `TSShapeConstructor`
- `getOffset()` : `Camera`
- `getOrbitDistance()` : `GuiObjectView`
- `getPacketLoss()` : `NetConnection`
- `getParameter()` : `SFXSource`
- `getParameterCount()` : `SFXSource`
- `getParameterName()` : `SFXParameter`
- `getParent()` : `GuiControl, GuiTreeViewCtrl`
- `getPathId()` : `Path`
- `getPendingImage()` : `ShapeBase`
- `getPing()` : `NetConnection`
- `getPitch()` : `SFXSource`
- `getPose()` : `Player`
- `getPosition()` : `SFXSound, SceneObject, GuiControl, Camera`,


StreamObject

- getPrevSibling(): GuiTreeViewCtrl
- getRandome(): SimSet
- getRechargeRate(): ShapeBase
- getRenderer(): GFXCardProfilerAPI
- getRepairRate(): ShapeBase
- getRightVector(): SceneObject
- getRoot(): GuiControl
- getRotation(): Camera
- getRowCount(): GuiFrameSetCtrl, GuiGameListMenuCtrl
- getRowId(): GuiTextListCtrl
- getRowLabel(): GuiGameListMenuCtrl
- getRowNumById(): GuiTextListCtrl
- getRowOffset(): GuiFrameSetCtrl
- getRowText(): GuiTextListCtrl
- getRowTextById(): GuiTextListCtrl
- getScale(): ActionMap, SceneObject
- getScrollPosition(): GuiScrollCtrl
- getScrollPositionX(): GuiScrollCtrl
- getScrollPositionY(): GuiScrollCtrl
- getSelCount(): GuiListBoxCtrl
- getSelected(): GuiPopUpMenuCtrlEx, GuiPopUpMenuCtrl
- getSelectedFile(): GuiDirectoryFileListCtrl
- getSelectedFiles(): GuiDirectoryFileListCtrl
- getSelectedId(): GuiTextListCtrl
- getSelectedItem(): GuiListBoxCtrl, GuiTreeViewCtrl
- getSelectedItemList(): GuiTreeViewCtrl
- getSelectedItems(): GuiListBoxCtrl
- getSelectedItemsCount(): GuiTreeViewCtrl
- getSelectedObject(): GuiTreeViewCtrl
- getSelectedObjectList(): GuiTreeViewCtrl
- getSelectedPage(): GuiTabBookCtrl
- getSelectedRow(): GuiGameListMenuCtrl, GuiTextListCtrl
- getSequenceBlend(): TSShapeConstructor
- getSequenceCount(): TSShapeConstructor
- getSequenceCyclic(): TSShapeConstructor
- getSequenceFrameCount(): TSShapeConstructor
- getSequenceGroundSpeed(): TSShapeConstructor
- getSequenceIndex(): TSShapeConstructor
- `getSequenceName()`: TSShapeConstructor
- `getSequencePriority()`: TSShapeConstructor
- `getSequenceSource()`: TSShapeConstructor
- `getServerConnection()`: GameConnection
- `getServerObject()`: NetObject
- `getShapeName()`: ShapeBase
- `getSkin()`: GuiObjectView
- `getSkinName()`: ShapeBase
- `getSlotTransform()`: ShapeBase
- `getSoundDuration()`: SFXProfile
- `getSource()`: SFXEmitter
- `getState()`: TurretShape, Player
- `getStatus()`: SFXSource, StreamObject
- `getStreamSize()`: StreamObject
- `getString()`: LangTable
- `getStringWidth()`: GuiControlProfile
- `getSuperClassNamespace()`: SimObject
- `getTarget()`: AITurretShape
- `getTargetCount()`: TSStatic, ShapeBase, InteriorInstance, TSShapeConstructor
- `getTargetName()`: InteriorInstance, TSShapeConstructor, ShapeBase, TSStatic
- `getTargetObject`: AIClient
- `getText`: GuiPopUpMenuCtrl, SimXMLDocument, GuiMLTextCtrl, GuiTextEditCtrl, GuiPopUpMenuCtrlEx, GuiButtonBaseCtrl
- `getTextById()`: GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
- `getTextToRoot()`: GuiTreeViewCtrl
- `getTime()`: GuiClockHud
- `getTransform()`: SceneObject
- `getTrigger()`: TSShapeConstructor, AIConnection
- `getTriggerCount()`: TSShapeConstructor
- `getTurretEulerRotation()`: TurretShape
- `getType()`: Message, SceneObject
- `getValue()`: SceneObject
- `getVelocity()`: Camera, ShapeBase
- `getVendor()`: GFXCardProfilerAPI
- `getVersion()`: GFXCardProfilerAPI
• getVideoMemoryMB() : GFXCardProfilerAPI
• getVideoMode() : GuiCanvas
• getVolume() : SFXSource
• getWeaponLeadVelocity() : AI_TurretShape
• getWheelCount() : WheeledVehicle
• getWhiteOut() : GameConnection, ShapeBase
• getWindowPosition() : GuiCanvas
• getWorldBox() : SceneObject
• getWorldBoxCenter() : SceneObject
• getWorldToScreenScale() : GuiTSCtrl
• getZoneId() : Zone
• glow : Material
• glowIntensity : Precipitation
• gravityCoefficient : ParticleData
• gravityMod : ProjectileData, ItemData, PhysicalZone
• gravModifier : DebrisData
• gridElementSize : WaterPlane, WaterBlock
• gridSize : WaterPlane, WaterBlock, GroundCover
• GroundAlpha : fxFoliageReplicator
• groundImpactMinSpeed : PlayerData
• groundImpactShakeAmp : PlayerData
• groundImpactShakeDuration : PlayerData
• groundImpactShakeFalloff : PlayerData
• groundImpactShakeFreq : PlayerData
• groupNum : GuiButtonBaseCtrl
• gustFrequency : ForestWindEmitter
• gustStrength : ForestWindEmitter
• gustWobbleStrength : ForestWindEmitter
• gustYawAngle : ForestWindEmitter
• gustYawFrequency : ForestWindEmitter
• gyroDrag : HoverVehicleData
Here is a list of all class members with links to the classes they belong to:
- h -

- handleRenameObject() : GuiTreeVewCtrl
- hardImpactSound : RigidShapeData, VehicleData
- hardImpactSpeed : RigidShapeData, VehicleData
- hardSplashSoundVelocity : PlayerData, RigidShapeData, VehicleData
- hasBitmapArray : GuiControlProfile
- hasImageState() : ShapeBase
- hasMount : ForestWindEmitter
- hasTarget() : AI<TurretShape
- headingRate : TurretShapeData
- height : CloudLayer, SplashData, BasicClouds
- HFReference : SFXEnvironment
- hidden : SimObject
- hide() : PfxVis
- hideCursor() : GuiCanvas
- HideFoliage : fxFoliageReplicator
- hideHeader : GuiRolloutCtrl
- HideReplications : fxShapeReplicator
- hideSelection() : GuiTreeVewCtrl
- highResOnly : ParticleEmitterData
- historySize : GuiTextEditCtrl
- hitAreaLowerRight : GuiGameListMenuProfile
- hitAreaUpperLeft : GuiGameListMenuProfile
- hitPlayers : Precipitation
- hitVehicles : Precipitation
- horizMaxSpeed : PlayerData
- horizontalSurfaceForce : FlyingVehicleData
- horizResistFactor : PlayerData
- horizResistSpeed : PlayerData
- horizSizing : GuiControl
- horizStacking : GuiStackControl
- hotSpot : GuiCursor
- hotTrackCallback : GuiPopupMenuCtrlEx
- hoverHeight : FlyingVehicleData
- hovertime : GuiControl
- hScrollBar : GuiScrollCtrl
Here is a list of all class members with links to the classes they belong to:
- i -

- iconBitmap : GuiIconButtonCtrl
- iconLocation : GuiIconButtonCtrl
- iconOffset : GuiGameListMenuProfile
- ignoreNodeScale : TSShapeConstructor
- ignoreWater : DebrisData
- imageAnimPrefix : PlayerData, ShapeBaseImageData
- imageAnimPrefixFP : PlayerData, ShapeBaseImageData
- imageTrigger0 : Player
- imageTrigger1 : Player
- impactForce : ProjectileData
- impactHardSound : PlayerData
- impactMetalSound : PlayerData
- impactSnowSound : PlayerData
- impactSoftSound : PlayerData
- impactSoundId : Material
- impactWaterEasy : RigidShapeData, VehicleData, PlayerData
- impactWaterHard : RigidShapeData, VehicleData, PlayerData
- impactWaterMedium : RigidShapeData, VehicleData, PlayerData
- import() : TerrainBlock
- includedStates : SFXState
- includeLightmappedGeometryInShadow : Sun, LightBase, LightDescription, ScatterSky
- increment : GuiTextEditSliderBitmapCtrl, GuiTextEditSliderCtrl
- indoorWeight : SpawnSphere
- inheritedVelFactor : ParticleData
- inheritEnergyFromMount : ShapeBaseData
- init() : Debris
- initialControlSet() : GameConnection
- initialPosition : Projectile
- initialVelocity : Projectile
- innerAngle : SpotLight
- InnerRadiusX : fxFoliageReplicator, fxShapeReplicator
- InnerRadiusY : fxFoliageReplicator, fxShapeReplicator
- insert() : ArrayObject
- insertItem() : GuiListBoxCtrl, GuiTreeViewCtrl
- insertLine() : MessageVector
- instantCollapse() : GuiRolloutCtrl
- instantExpand() : GuiRolloutCtrl
- integration : RigidShapeData, VehicleData
- Interactions : fxShapeReplicator
- interiorFile : InteriorInstance
- internalName : SimObject
- invertLayer : GroundCover
- invisible : GuiSeparatorCtrl
- is3D : SFXDescription, SFXEmitter
- isActive : GuiControl, SFXState
- isAIControlled : ShapeBase, GameConnection
- isAllTextSelected() : GuiTextEditCtrl
- isAtRest() : Item
- isAttached() : PhysicsForce
- isAutoGenerated : Material
- isAwake() : GuiControl
- isBallistic : ProjectileData
- isBanned() : BanList
- isChildOfGroup() : SimObject
- isClientObject() : NetObject
- isCloaked() : ShapeBase
- isContainer : GuiControl
- isControlObjectRotDampedCamera() : GameConnection
- isCursorOn() : GuiCanvas
- isCursorShown() : GuiCanvas
- isDataLoaded() : SFXFMODEventGroup
- isDemoPlaying() : GameConnection
- isDemoRecording() : GameConnection
- isDestroyed() : ShapeBase, PhysicsShape
- isDirty() : Forest
- isDisabled() : ShapeBase, SFXState
- isEditOrbitMode() : Camera
- isEditorOnly() : SimObject
- isEnabled : LightBase, PostEffect, ShapeBase, PostEffect
- isEOF() : FileObject, StreamObject
- isEOS() : StreamObject
- isExpanded() : SimObject, GuiRolloutCtrl
- isExteriorPortal() : Portal
- isField() : SimObject
isFirstPerson() : GameConnection
isFirstResponder() : GuiControl
isFrozen() : GuiStackControl
isFullscreen() : GuiCanvas
isGlobalBounds() : SceneObject
isHidden() : ShapeBase
isImageFiring() : ShapeBase
isImageMounted() : ShapeBase
isInNamespaceHierarchy() : SimObject
isInteriorPortal() : Portal
isInverted() : ActionMap
isInvincible : ShapeBaseData
isItemSelected() : GuiTreeViewCtrl
isLooping : SFXDescription, SFXEmitter, GuiAutoScrollCtrl, Path
isMaximized() : GuiCanvas
isMember() : SimSet
isMemberOfClass() : SimObject
isMethod() : SimObject
isMinimized() : GuiCanvas
isMounted() : SceneObject
isMouseLocked() : GuiControl
isNameChangeAllowed() : SimObject
isObjectMirrored() : GuiListBoxCtrl
isParentItem() : GuiTreeViewCtrl
isPaused() : SFXSource
isPlaybackDone() : GuiTheoraCtrl
isPlaying() : SFXSource
isReady() : SFXSound
isRegisteredEvent() : EventManager
isRenderable : Camera, SceneObject, ConvexShape, BasicClouds, GameBase, ShapeBase, MissionMarker, PxCloth, WayPoint, OcclusionVolume, LightBase, GroundCover, PointLight, StaticShape, Spotlight, DecalManager, RenderObjectExample, RenderShapeExample, Explosion, fxFoliageReplicator, fxShapeReplicator, fxShapeReplicatedStatic, ParticleEmitter, Splash, PhysicsDebris, SFXSpace, Vehicle, HoverVehicle, Forest, CloudLayer, WaterObject, WaterBlock, WaterPlane, Marker,
Player, AIPlayer, Debris, GroundPlane, Item, PathCamera, PhysicalZone, Zone, Prefab, Projectile, ProximityMine, RigidShape, Trigger, TSStatic, RenderMeshExample, InteriorInstance, ParticleEmitterNode, Precipitation, PhysicsForce, PxMultiActor, SFXEmitter, TurretShape, Sun, FlyingVehicle, WheeledVehicle, DecalRoad, River, ScatterSky, SkyBox, MeshRoad, TimeOfDay, ForestWindEmitter, TerrainBlock, ALTurretShape, PhysicsShape, Lightning, Portal, SpawnSphere

- isRenderEnabled: SceneObject
- isRotating(): Item
- isRotationDamped(): Camera
- isRowActive(): GuiTextListCtrl
- isRowEnabled(): GuiGameListMenuCtrl
- isSelectable: Precipitation, CloudLayer, PhysicsShape, MeshRoad, OcclusionVolume, WayPoint, fxFoliageReplicator, GroundCover, Lightning, River, PhysicsDebris, PointLight, ParticleEmitterNode, RigidShape, RenderMeshExample, SpotLight, SFXEmitter, fxShapeReplicatedStatic, LightBase, SceneObject, PathCamera, Sun, GameBase, ShapeBase, PxMultiActor, Vehicle, ScatterSky, Player, SkyBox, WaterObject, ForestWindEmitter, DecalManager, WheeledVehicle, RenderObjectExample, Marker, InteriorInstance, Splash, Forest, PhysicsForce, ALTurretShape, BasicClouds, WaterBlock, HoverVehicle, ConvexShape, SpawnSphere, Trigger, Projectile, RenderShapeExample, TSStatic, ParticleEmitter, StaticShape, GroundPlane, PhysicalZone, TimedOfDay, fxShapeReplicator, ProximityMine, Item, Prefab, Portal, Explosion, Debris, TurretShape, DecalRoad, AIPlayer, Camera, PxCloth, FlyingVehicle, Zone, WaterPlane, TerrainBlock, MissionMarker, SFXSpace

- isSelected(): SimObject
- isSelectionEnabled: SceneObject
- isServerObject(): NetObject
- isStateOn(): GuiCheckBoxCtrl
- isStatic(): Item
- isStopped(): SFXSource
- isStreaming: SFXEmitter, SFXDescription
• isThumbBeingDragged() : GuiSliderCtrl
• isValidDragTarget() : GuiTreeViewCtrl
• isVisible() : GuiControl
• itemHeight : GuiTreeViewCtrl
Here is a list of all class members with links to the classes they belong to:
- j -

- jetEnergyDrain : VehicleData
- jetForce : VehicleData
- jetJumpEnergyDrain : PlayerData
- jetJumpForce : PlayerData
- jetJumpSurfaceAngle : PlayerData
- jetMaxJumpSpeed : PlayerData
- jetMinJumpEnergy : PlayerData
- jetMinJumpSpeed : PlayerData
- jetSound : FlyingVehicleData, HoverVehicleData, WheeledVehicleData
- jumpDelay : PlayerData
- jumpEnergyDrain : PlayerData
- jumpForce : PlayerData
- jumpJetTrigger : Player
- jumpSurfaceAngle : PlayerData
- jumpTowardsNormal : PlayerData
- jumpTrigger : Player
- justify : GuiControlProfile
Here is a list of all class members with links to the classes they belong to:
- k -

- key : ArrayObject
- kineticFriction : WheeledVehicleTire
Here is a list of all class members with links to the classes they belong to:
- I -

- landSequenceTime : PlayerData
- langTableMod : GuiControl
- lastAttribute() : SimXMLDocument
- lastSplitTerrainOnly : LightBase, LightDescription, ScatterSky, Sun
- lateralDamping : WheeledVehicleTire
- lateralForce : WheeledVehicleTire
- lateralRelaxation : WheeledVehicleTire
- layer : GroundCover
- layerEnabled : BasicClouds
- leaveCommand : Trigger
- leftMargin : GuiSeparatorCtrl
- length : WheeledVehicleSpring, GuiSpeedometerHud
- level : ConsoleLogger
- LFReference : SFXEnvironment
- lifeSpan : DecalData
- lifetime : DebrisData, Debris, ProjectileData, PhysicsDebrisData
- lifetimeMS : ExplosionData, SplashData, ParticleData, ParticleEmitterData
- lifetimeVariance : DebrisData, ExplosionData, SplashData, PhysicsDebrisData
- lifetimeVarianceMS : ParticleData, ParticleEmitterData
- lightAmbient : GuiObjectView
- lightBrightness : ShapeBaseImageData
- lightColor : ItemData, GuiObjectView, ShapeBaseImageData
- lightDesc : ProjectileData
- lightDirection : GuiObjectView
- lightDuration : ShapeBaseImageData
- lightEndBrightness : ExplosionData
- lightEndColor : ExplosionData
- lightEndRadius : ExplosionData
- lightMap : Material
- lightMapSize : TerrainBlock
- lightNormalOffset : ExplosionData
- LightOn : fxFoliageReplicator
- lightOnlyStatic : ItemData
• lightRadius : ItemData, ShapeBaseImageData
• lightStartBrightness : ExplosionData
• lightStartColor : ExplosionData
• lightStartRadius : ExplosionData
• LightSync : fxFoliageReplicator
• lightTime : ItemData, fxFoliageReplicator
• lightType : ShapeBaseImageData, ItemData
• linearDamping : PhysicsShapeData, PhysicsDebrisData
• linearDrag : PxMultiActorData
• linearSleepThreshold : PhysicsDebrisData, PhysicsShapeData
• lineContinuedIndex : GuiMessageVectorCtrl
• lineSpacing : GuiMLTextCtrl, GuiMessageVectorCtrl
• liquidType : WaterObject
• listClassIDs() : GameConnection
• listen() : TCPObject
• listMeshes() : PxMultiActor
• listObjects() : SimSet
• loadData() : SFXFMODEventGroup
• loadFile() : SimXMLDocument
• locked : SimObject
• lockFrustum : GroundCover
• lockHorizScroll : GuiScrollCtrl
• lockMouse : GuiMouseEventCtrl
• lockSelection() : GuiTreeViewCtrl
• lockVertScroll : GuiScrollCtrl
• lodReflectScalar : Forest
• lodType : TSShapeConstructor
• logWeight : LightBase, ScatterSky, LightDescription, Sun
• longitudinalDamping : WheeledVehicleTire
• longitudinalForce : WheeledVehicleTire
• longitudinalRelaxation : WheeledVehicleTire
• lookAt() : Camera
• loopMode : SFXPlayList
• LowLODDistance : River
Here is a list of all class members with links to the classes they belong to:
- m -

- magFilter : GFXSamplerStateData
- mainThrustForce : HoverVehicleData
- makeFirstResponder() : GuiControl
- makeIconSquare : GuiIconButtonCtrl
- makeNameCallback : GuiListBoxCtrl
- maneuveringForce : FlyingVehicleData
- mapTo : Material
- margin : GuiContainer, GuiRolloutCtrl
- markerName : WayPoint
- markItem() : GuiTreeViewCtrl
- mass : Camera, ShapeBaseImageData, PhysicsDebrisData, ShapeBaseData, PhysicsShapeData, WheeledVehicleTire, ForestItemData
- massBox : RigidShapeData, VehicleData
- massCenter : RigidShapeData, VehicleData
- matchColor : GuiMessageVectorCtrl
- matchScale() : GuiGraphCtrl
- matchVideoSize : GuiTheoraCtrl
- Material : PxMultiActorData, DecalRoad, SkyBox, ConvexShape, GroundCover, PxCloth, GroundPlane, DecalData, RenderMeshExample
- matNamePrefix : TSShapeConstructor
- maxAngle : GuiSpeedometerHud
- maxAnisotropy : GFXSamplerStateData
- maxAutoSpeed : FlyingVehicleData
- maxBackwardSpeed : PlayerData
- maxBillboardTiltAngle : GroundCover
- maxChars : GuiMLTextCtrl
- maxClumpCount : GroundCover
- maxColorIndex : GuiMessageVectorCtrl
- maxConcurrentSounds : ShapeBaseImageData
- maxCrouchBackwardSpeed : PlayerData
- maxCrouchForwardSpeed : PlayerData
- maxCrouchSideSpeed : PlayerData
- maxDamage : ShapeBaseData
- maxDistance : SFXDescription, SFXPlayList, SFXEmitter
- maxDistanceVariance : SFXPlayList
- maxDrag: RigidShapeData, VehicleData
- maxElements: GroundCover
- maxElevation: GroundCover
- maxEnergy: ShapeBaseData
- maxForwardSpeed: PlayerData
- maxFreelookAngle: PlayerData
- maxHeading: TurretShapeData
- MaxHeight: fxFoliageReplicator
- maximizeWindow(): GuiCanvas
- maxImpulseVelocity: Player
- maxJumpSpeed: PlayerData
- maxLength: GuiTextCtrl
- maxLookAngle: PlayerData
- MaxLuminance: fxFoliageReplicator
- maxMass: Precipitation
- maxOrbitDistance: GuiObjectView
- maxPitch: TurretShapeData
- maxPopupHeight: GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
- maxPredictionTicks: Player
- maxProneBackwardSpeed: PlayerData
- maxProneForwardSpeed: PlayerData
- maxProneSideSpeed: PlayerData
- maxRateMs: ReflectorDesc
- maxScanDistance: AITurretShapeData
- maxScanHeading: AITurretShapeData
- maxScanPitch: AITurretShapeData
- maxSideSpeed: PlayerData
- maxSlope: GroundCover
- maxSpeed: GuiSpeedometerHud, Precipitation
- maxSpinSpeed: DebrisData
- maxSprintBackwardSpeed: PlayerData
- maxSprintForwardSpeed: PlayerData
- maxSprintSideSpeed: PlayerData
- maxSteeringAngle: VehicleData
- maxStepHeight: PlayerData
- MaxSwayTime: fxFoliageReplicator
- maxTimeScale: PlayerData
- maxTurbulence: Precipitation
- maxUnderwaterBackwardSpeed: PlayerData
- maxUnderwaterForwardSpeed : PlayerData
- maxUnderwaterSideSpeed : PlayerData
- maxVelocity : ItemData
- maxWarpTicks : Player, Item
- maxWeaponRange : AI_TurretShapeData
- maxWheelSpeed : WheeledVehicleData
- MaxWidth : fxFoliageReplicator
- mediaPath : SFXFMODProject
- mediumSplashSoundVelocity : RigidShapeData, VehicleData, PlayerData
- meshCulling : fxFShapeReplicatedStatic, TSStatic
- mieScattering : ScatterSky
- minAngle : Gui_Speedometer_Hud
- minClumpCount : GroundCover
- minDrag : RigidShapeData, VehicleData
- minElevation : GroundCover
- minEnergy : ShapeBaseImageData
- minExtent : GuiControl
- minFilter : GFXSamplerStateData
- MinHeight : fxFoliageReplicator
- minimizeWindow() : GuiCanvas
- minImpactSpeed : RigidShapeData, VehicleData, PlayerData
- minJetEnergy : VehicleData
- minJumpEnergy : PlayerData
- minJumpSpeed : PlayerData
- minLaterallImpactSpeed : PlayerData
- minLookAngle : PlayerData
- MinLuminance : fxFoliageReplicator
- minMass : Precipitation
- minNaertConstant : Material
- minOrbitDiststance : GuiObjectView
- minPitch : TurretShapeData
- minRollSpeed : RigidShapeData, VehicleData
- minRunEnergy : PlayerData
- minSpeed : Precipitation
- minSpinSpeed : DebrisData
- minSprintEnergy : PlayerData
- MinSwayTime : fxFoliageReplicator
- minTabWidth : GuiTabBookCtrl
- `minTrailSpeed`: FlyingVehicleData
- `minWarpTicks`: Player, Item
- `MinWidth`: fxFoliageReplicator
- `mipFilter`: GFXSamplerStateData
- `mipLODBias`: GFXSamplerStateData
- `mirrorSet`: GuiListBoxCtrl
- `missionCycleCleanup`: AIClient
- `mMoveTolerance`: AIPlayer
- `modal`: GuiControlProfile, GuiControl
- `modifyStorm()`: Precipitation
- `modulationDepth`: SFXEnvironment
- `modulationTime`: SFXEnvironment
- `moonAzimuth`: ScatterSky
- `moonElevation`: ScatterSky
- `moonEnabled`: ScatterSky
- `moonLightColor`: ScatterSky
- `moonMat`: ScatterSky
- `moonScale`: ScatterSky
- `mountedNode`: GuiObjectView
- `mountedShapeFile`: GuiObjectView
- `mountedSkin`: GuiObjectView
- `mountImage()`: ShapeBase
- `mountNode`: SceneObject
- `mountObject()`: SceneObject
- `mountPID`: SceneObject
- `mountPoint`: ShapeBaseImageData
- `mountPos`: SceneObject
- `mountRot`: SceneObject
- `mouseDragging`: GuiTreeViewCtrl
- `mouseOverSelected`: GuiControlProfile
- `mouseWheelScrollSpeed`: GuiScrollCtrl
- `move`: AIClient
- `moveFirst()`: ArrayObject
- `moveForward`: AIClient
- `moveItemDown()`: GuiTreeViewCtrl
- `moveItemUp()`: GuiTreeViewCtrl
- `moveLast()`: ArrayObject
- `movementSpeed`: Camera
- `moveNext()`: ArrayObject
- movePrev() : ArrayObject
- moveStuckTestDelay : AIPlayer
- moveStuckTolerance : AIPlayer
- movingBubblesSound : PlayerData
- msg() : SimpleMessageEvent
- msToNext : Marker
- MultipleFiles : OpenFileDialog
- multipleSelections : GuiTreeViewCtrl
- MustExist : OpenFileDialog
- muzzleVelocity : ProjectileData
Here is a list of all class members with links to the classes they belong to:
- n -

- name : SimObject
- nearClip : LevelInfo
- nearDist : ReflectorDesc
- neverImport : TSShapeConstructor
- neverImportMesh : TSShapeConstructor
- newtonMode : Camera
- newtonRotation : Camera
- nextAttribute() : SimXMLDocument
- nextSiblingElement() : SimXMLDocument
- nightColor : ScatterSky
- nightCubemap : ScatterSky
- nightFogColor : ScatterSky
- nightScale : TimeOfDay
- noBillboards : GroundCover
- noCorrection : PxMultiActorData
- noCorrections : PhysicsShape
- Node : DecalRoad, MeshRoad, River
- noIndividualDamage : StaticShapeData
- normalForce : HoverVehicleData
- normalMap : TerrainMaterial, Material
- noShapes : GroundCover
- noSmoothing : PhysicsShape
- notifyShapeChanged() : TSShapeConstructor
- numbersOnly : GuiControlProfile
- numBounces : DebrisData
- numDmgEmitterAreas : VehicleData
- numDrops : Precipitation
- numFences : GuiCanvas
- numSegments : SplashData
- numSlotsToPlay : SFXPlayList
- numSplits : LightDescription, LightBase, ScatterSky, Sun
Here is a list of all class members with links to the classes they belong to:
- o -

- objectTypeMask : Reflectordesc
- observeThroughObject : ShapeBaseData
- occlusionRadius : LightFlareData
- offset : ExplosionData, ShapeBaseImageData
- offsetA : LightAnimData
- offsetKeys : LightAnimData
- offsetPeriod : LightAnimData
- offsetSmooth : LightAnimData
- OffsetZ : LightAnimData, fxFoliageReplicator, fxShapeReplicator
- OGLPixelShaderFile : ShaderData
- OGLVertexShaderFile : ShaderData
- onAction() : GuiControl
- onActivate() : SFXState
- onActive() : GuiControl
- onAdd() : GameBaseData, GuiControl, ScriptMsgListener, SpawnSphere, Trigger, ScriptObject, ScriptGroup, PostEffect, Message
- onAddGroupSelected() : GuiTreeCtrl
- onAddMultipleSelectionBegin() : GuiTreeCtrl
- onAddMultipleSelectionEnd() : GuiTreeCtrl
- onAddSelection() : GuiTreeCtrl
- onAddToQueue() : ScriptMsgListener
- onAltClick() : GuiBitmapButtonCtrl
- onBeginReparenting() : GuiTreeCtrl
- onChange() : GuiGameListMenuCtrl
- onChildAdded : GuiScriptNotifyCtrl
- onChildRemoved : GuiScriptNotifyCtrl
- onChildResized : GuiScriptNotifyCtrl
- onClearSelection() : GuiListBoxCtrl, GuiTreeCtrl
- onClick() : GuiButtonBaseCtrl
- onClose() : GuiWindowCtrl
- onCollapse() : GuiWindowCtrl
- onCollapsed() : GuiRolloutCtrl
- onCollision() : ShapeBaseData, ProjectileData
- onComplete() : GuiAutoScrollCtrl
- onConnected() : TCPObject
- onConnectFailed(): TCPObj
- onConnectionAccepted(): GameConnection
- onConnectionDropped(): GameConnection
- onConnectionError(): GameConnection
- onConnectionRequest(): TCPObj
- onConnectionTimedOut(): GameConnection
- onConnectRequestRejected(): GameConnection
- onConnectRequestTimedOut(): GameConnection
- onControlDragEnter(): GuiControl
- onControlDragExit(): GuiControl
- onControlDragged(): GuiControl
- onControlDropped(): GuiControl
- onControlObjectChange(): GameConnection
- onCtrlClick(): GuiBitmapButtonCtrl
- onDamage(): ShapeBaseData
- onDataBlocksDone(): GameConnection
- onDeactivate(): SFXState
- onDefaultClick(): GuiBitmapButtonCtrl
- onDefineIcons(): GuiTreeViewCtrl
- onDeleteKey(): GuiListBoxCtrl, GuiTextListCtrl
- onDeleteObject(): GuiTreeViewCtrl
- onDeleteSelection(): GuiTreeViewCtrl
- onDestroyed(): ShapeBaseData
- onDialogPop(): GuiControl
- onDialogPush(): GuiControl
- onDisabled(): ShapeBaseData, PostEffect
- onDisconnect(): TCPObj
- onDNSFailed(): TCPObj
- onDNSResolved(): TCPObj
- onDone(): GuiFadeinBitmapCtrl
- onDoubleClick(): GuiButtonBaseCtrl, GuiListBoxCtrl
- onDragDropped(): GuiTreeViewCtrl
- onDrop(): GameConnection
- oneFrameOnly: PostEffect
- onEnabled(): ShapeBaseData, PostEffect
- onEndReparenting(): GuiTreeViewCtrl
- onEndSequence(): ShapeBaseData
- onEnterLiquid(): VehicleData, Item, PlayerData, RigidShape
- onEnterMissionArea(): PlayerData
onEnterTrigger() : TriggerData
onExpanded() : GuiRolloutCtrl
onExplose() : ProjectileData, ProximityMineData
onFlash() : GameConnection
onForceUncloak() : ShapeBaseData
onGainFirstResponder : GuiScriptNotifyCtrl, GuiControl, GuiScriptNotifyCtrl
onHeaderRightClick() : GuiRolloutCtrl
onImpact() : ShapeBaseData
onInputEvent() : GuiInputCtrl
onInspect() : GuiTreeViewCtrl
onKeyDown() : GuiTreeViewCtrl
onLeaveLiquid() : VehicleData, Item, PlayerData, RigidShape
onLeaveMissionArea() : PlayerData
onLeaveTrigger() : TriggerData
onLine() : TCPObject
onLoad() : Prefab, TSShapeConstructor
onLoseFirstResponder : GuiScriptNotifyCtrl, GuiControl, GuiScriptNotifyCtrl
onMaximize() : GuiWindowCtrl
onMenuItemSelect() : GuiMenuBar
onMenuSelect() : GuiMenuBar
onMessageObjectReceived() : ScriptMsgListener
onMessageReceived() : ScriptMsgListener
onMessageSelected() : GuiConsole
onMinimize() : GuiWindowCtrl
onMount() : GameBaseData, ShapeBaseImageData
onMountObject() : TurretShapeData
onMouseDown() : GuiButtonBaseCtrl, GuiMouseEventCtrl
onMouseDragged() : GuiListBoxCtrl, GuiTreeCtrl, GuiLObjectView, GuiMouseEventCtrl
onMouseEnter() : GuiButtonBaseCtrl, GuiObjectView, GuiMouseEventCtrl
onMouseInMenu() : GuiMenuBar
onMouseLeave() : GuiButtonBaseCtrl, GuiObjectView, GuiMouseEventCtrl
onMouseMove() : GuiMouseEvCtrl
onMouseUp() : GuiButtonBaseCtrl, GuiListBoxCtrl, GuiTreeCtrl, GuiMouseEvCtrl
- onNewDataBlock() : GameBaseData
- onNode() : PathCamera
- onObjectAdded() : SimSet
- onObjectDeleteCompleted() : GuiTreeViewCtrl
- onObjectRemoved() : SimSet
- onParameterValueChange() : SFXSource
- onParentResized() : GuiScriptNotifyCtrl
- onPoseChange() : PlayerData
- onRemove() : GameBaseData, GuiControl, ScriptMsgListener, Trigger, ScriptObject, ScriptGroup, Message
- onRemoveFromQueue() : ScriptMsgListener
- onRemoveSelection() : GuiTreeViewCtrl
- onRenameValidate : GuiTreeViewCtrl
- onReparent() : GuiTreeViewCtrl
- onReset() : GuiAutoScrollCtrl
- onResize() : GuiMLTextCtrl, GuiScriptNotifyCtrl
- onRestore() : GuiWindowCtrl
- onReturn() : GuiTextEditCtrl
- onRightClick() : GuiButtonBaseCtrl
- onRightMouseDown() : GuiMouseEventCtrl, GuiTreeViewCtrl
- onRightMouseDragged() : GuiMouseEventCtrl
- onRightMouseUp() : GuiTreeViewCtrl, GuiMouseEventCtrl
- onScroll() : GuiScrollCtrl
- onSelect() : GuiListBoxCtrl, GuiTreeViewCtrl, GuiTextListCtrl
- onShiftClick() : GuiBitmapButtonCtrl
- onSleep() : GuiControl
- onStart() : GuiAutoScrollCtrl
- onStartSprintMotion() : PlayerData
- onStartSwim() : PlayerData
- onStatusChange() : SFXSource
- onStickyCollision() : TurretShapeData, Item
- onStopSprintMotion() : PlayerData
- onStopSwim() : PlayerData
- onSubmenuSelect() : GuiMenuBar
- onTabComplete() : GuiTextEditCtrl
- onTabRightClick() : GuiTabBookCtrl
- onTabSelected() : GuiTabBookCtrl
- onThisFrame : PostEffect
- onTick() : GuiAutoScrollCtrl
- `onTickTrigger()` : `TriggerData`
- `onTrigger()` : `ShapeBaseData`
- `onTriggered()` : `ProximityMineData`
- `onUnload()` : `TSShapeConstructor`
- `onUnmount()` : `ShapeBaseImageData`, `GameBaseData`
- `onUnmountObject()` : `TurretShapeData`
- `onUnselect()` : `GuiTreeViewCtrl`, `GuiListBoxCtrl`
- `onUpdate()` : `SFXParameter`
- `onURL()` : `GuiMLTextCtrl`
- `onValidate()` : `GuiTextEditCtrl`
- `onVisible()` : `GuiControl`
- `onWake()` : `GuiControl`
- `onWindowClosed()` : `PfxVis`
- `opaque` : `GuiControlProfile`
- `open()` : `PfxVis`, `FileStreamObject`, `GuiTreeViewCtrl`
- `openArchive()` : `ZipObject`
- `openFileForRead()` : `ZipObject`
- `openFileForWrite()` : `ZipObject`
- `openForAppend()` : `FileObject`
- `openForRead()` : `FileObject`
- `openForWrite()` : `FileObject`
- `orbitDistance` : `GuiObjectView`
- `orientation` : `GuiSplitContainer`
- `orientOnVelocity` : `ParticleEmitterData`
- `orientParticles` : `ParticleEmitterData`
- `originSort` : `fxShapeReplicatedStatic`, `TSStatic`
- `outdoorWeight` : `SpawnSphere`
- `outerAngle` : `SpotLight`
- `OuterRadiusX` : `fxShapeReplicator`, `fxFoliageReplicator`
- `OuterRadiusY` : `fxShapeReplicator`, `fxFoliageReplicator`
- `overallFoamOpacity` : `WaterObject`
- `overallRippleMagnitude` : `WaterObject`
- `overallScale` : `LightFlareData`
- `overallWaveMagnitude` : `WaterObject`
- `overDarkFactor` : `LightBase`, `Sun`, `LightDescription`, `ScatterSky`
- `overlayMap` : `Material`
- `overlayTex` : `Material`
- `overrideAdvance` : `ParticleEmitterData`
- `OverwritePrompt` : `SaveFileDialog`
Here is a list of all class members with links to the classes they belong to:
- p -

- padding : GuiContainer, GuiDynamicCtrlArrayControl, GuiMenuBar, GuiStackControl
- parallaxScale : Material, TerrainMaterial
- parameters : SFXTrack, SFXDescription
- parentGroup : SimObject
- parse() : SimXMLDocument
- particleDensity : ExplosionData
- ParticleEmitter : ExplosionData, ProjectileData
- particleRadius : ExplosionData
- particles : ParticleEmitterData
- particleWaterEmitter : ProjectileData
- password : GuiTextEditCtrl
- passwordMask : GuiTextEditCtrl
- pause() : SFXSource, GuiTheoraCtrl
- pauseAnimation : LightBase
- pauseThread() : ShapeBase
- peekLine() : FileObject
- performClick() : GuiButtonBaseCtrl
- periodVarianceMS : ParticleEmitterData
- persistentId : SimObject
- phiReferenceVel : ParticleEmitterData
- phiVariance : ParticleEmitterData
- physicsPlayerType : PlayerData
- physXStream : PxMultiActorData
- pickupRadius : PlayerData
- pitch : SFXDescription, SFXEmitter
- pitchForce : HoverVehicleData
- pitchRate : TurretShapeData
- pitchScale : SFXPlayList
- pitchScaleVariance : SFXPlayList
- pixelSpecular : Material
- pixVersion : ShaderData
- PlacementAreaHeight : fxFoliageReplicator, fxShapeReplicator
- PlacementColour : fxFoliageReplicator, fxShapeReplicator
- planarReflection : Material
- plane : OcclusionVolume, SFXSpace, Zone
- play : TimeOfDay, SFXEmitter, GuiTheoraCtrl
- play2D() : GameConnection
- play3D() : GameConnection
- playAmbient : fxShapeReplicatedStatic, TSStatic, PhysicsShape
- playAnimation() : LightBase
- playAudio() : ShapeBase
- playDemo() : GameConnection
- playOnAdd : SFXEmitter
- playOnWake : GuiTheoraCtrl
- playSpeed : ExplosionData
- playThread() : ShapeBase
- plotColor : GuiGraphCtrl
- plotInterval : GuiGraphCtrl
- plotType : GuiGraphCtrl
- plotVariable : GuiGraphCtrl
- point : OcclusionVolume, SFXSpace, Zone
- pointInControl() : GuiControl
- polyhedron : PhysicalZone, Trigger
- pop() : ActionMap
- pop_back() : ArrayObject
- pop_front() : ArrayObject
- popBackLine() : MessageVector
- popDialog() : GuiCanvas
- popElement() : SimXMLDocument
- popFront() : PathCamera
- popFrontLine() : MessageVector
- popLayer() : GuiCanvas
- position : SceneObject, GuiControl
- post() : HTTPObject
- postApply() : MissionArea, MeshRoad, DecalData, SkyBox, GroundPlane, RenderMeshExample, DecalRoad
- postEvent() : EventManager
- preload : PhysicsDebrisData, SFXProfile
- preProcess() : PostEffect
- presimulate() : Projectile
- prevAttribute() : SimXMLDocument
- priority : LightBase, ReflectorDesc, SFXDescription
- probability : GroundCover, ForestBrushElement
- processAddOrder : RenderBinManager
- profile : GuiControl
- profileForChildren : GuiControlProfile
- project() : GuiTSCtrl
- Projectile : ShapeBaseImageData
- projectileShapeName : ProjectileData
- proneBoundingBox : PlayerData
- proneForce : PlayerData
- proneTrigger : Player
- pulseRate : GuiHealthBarHud, GuiHealthTextHud
- pulseThreshold : GuiHealthTextHud, GuiHealthBarHud
- push() : ActionMap
- push_back() : ArrayObject
- push_front() : ArrayObject
- pushBack() : PathCamera
- pushBackLine() : MessageVector
- pushChildElement() : SimXMLDocument
- pushDialog() : GuiCanvas
- pushFirstChildElement() : SimXMLDocument
- pushFront() : PathCamera
- pushFrontLine() : MessageVector
- pushNewElement() : SimXMLDocument
- pushToBack() : SimSet

Copyright © GarageGames, LLC. All Rights Reserved.
Here is a list of all class members with links to the classes they belong to:
- q -

- queryProfile() : GFXCardProfilerAPI
- queue : EventManager

Copyright © GarageGames, LLC. All Rights Reserved.
Here is a list of all class members with links to the classes they belong to:
- r -

- radialEmitter : ForestWindEmitter
- radius : PointLight, WheeledVehicleTire, ForestItemData, GroundCover, SpawnSphere, ForestWindEmitter
- random : SFXPlayList
- RandomFlip : fxFoliageReplicator
- randomize : DecalData
- range : SFXParameter, SpotLight, GuiTextEditSliderBitmapCtrl, GuiTextEditSliderCtrl, LightDescription, GuiSliderCtrl
- rayleighScattering : ScatterSky
- readComment() : SimXMLDocument
- readLine() : FileObject, StreamObject
- readLongString() : StreamObject
- readString() : StreamObject
- readSTString() : StreamObject
- recenterTurret() : AITurretShape
- recoverDelay : PlayerData
- recoverRunForceScale : PlayerData
- referenceDistance : SFXDescription, SFXPlayList, SFXEmitter
- referenceDistanceVariance : SFXPlayList
- reflect : Precipitation
- reflectDetailAdjust : WaterObject
- reflections : SFXEnvironment
- reflectionsDelay : SFXEnvironment
- reflectionsPan : SFXEnvironment
- reflectivity : WaterObject
- reflectMaxRateMs : WaterObject
- reflectNormalUp : WaterObject
- reflectPriority : GuiTSCtrl, WaterObject
- reflectScale : GroundCover
- reflectTexSize : WaterObject
- refresh() : GuiDynamicCtrlArrayControl
- regenCells() : Forest
- regenerate() : DecalRoad, MeshRoad, River
- registerEvent() : EventManager
- reload : Material, PxMultiActorData, ParticleData, ParticleEmitterData, GuiDirectoryFileListCtrl, ShaderData, PostEffect
- reloadOnLocalClient() : SimDataBlock
- remove() : SimSet, EventManager
- removeAll() : EventManager
- removeAllChildren : GuiTreeViewCtrl
- removeAutoPlot() : GuiGraphCtrl
- removeBan() : BanList
- removeChildSelectionByValue : GuiTreeViewCtrl
- removeColumn() : GuiFrameSetCtrl
- removeDetailLevel() : TSShapeConstructor
- removeFilteredItem() : GuiListBoxCtrl
- removeFromIgnoreList() : AIAudio
- removeImposter() : TSShapeConstructor
- removeItem() : GuiTreeViewCtrl
- removeManager() : RenderPassManager
- removeMenu() : GuiMenuBar
- removeMenuItem() : GuiMenuBar
- removeMesh() : TSShapeConstructor
- removeNode() : TSShapeConstructor
- removeObject() : TSShapeConstructor
- removeParameter() : SFXSource
- removeRow() : GuiFrameSetCtrl, GuiTextListCtrl
- removeRowById() : GuiTextListCtrl
- removeSelection() : Gui(TreeViewCtrl
- removeSequence() : TSShapeConstructor
- removeShaderMacro() : PostEffect
- removeText() : SimXMLDocument
- removeTrigger() : TSShapeConstructor
- renameDetailLevel() : TSShapeConstructor
- renameInternal : Gui(TreeViewCtrl
- renameNode() : TSShapeConstructor
- renameObject() : TSShapeConstructor
- renameSequence() : TSShapeConstructor
- renderBin : PostEffect
- renderCells : GroundCover
- renderCollision : Player
- renderColorInnerCone : SFXEmitter
- renderColorOuterCone : SFXEmitter
- renderColorOutsideVolume : SFXEmitter
- renderColorPlayingInRange : SFXEmitter
- renderColorPlayingOutOfRange : SFXEmitter
- renderColorRangeSphere : SFXEmitter
- renderColorStoppedInRange : SFXEmitter
- renderColorStoppedOutOfRange : SFXEmitter
- renderDebugInfo : GuiTheoraCtrl
- renderedBatches : GroundCover
- renderedBillboards : GroundCover
- renderedCells : GroundCover
- renderedShapes : GroundCover
- renderEmitters : SFXEmitter
- renderFirstPerson : PlayerData
- renderFront() : GuiCanvas
- renderMyItems : Player
- renderMyPlayer : Player
- renderNormals : fxShapeReplicatedStatic, TSStatic
- renderOffset : GuiCursor
- renderOrder : RenderBinManager
- renderPointDistance : SFXEmitter
- renderPriority : DecalData, DecalRoad, PostEffect
- renderRadialIncrements : SFXEmitter
- renderReflection : ParticleEmitterData
- renderReflectPass : LightFlareData
- renderSweepIncrements : SFXEmitter
- renderTime : PostEffect
- renderTriggers : Trigger
- renderWhenDestroyed : ShapeBaseData
- renderWireframe : RenderTerrainMgr
- renderZones : PhysicalZone
- reorderChild() : SimSet
- repaint() : GuiCanvas
- repairRate : ShapeBaseData
- repeatCount : SFXPlayList
- replaceText : GuiPopUpMenuCtrlEx, GuiPopUpMenuCtrl
- replay : SFXPlayList
- representedInLightmap : LightDescription, ScatterSky, Sun, LightBase
- reset() : RigidShape, SimXMLDocument, GuiAutoScrollCtrl, SFXParameter, GuiCanvas, PathCamera
- resetDelay : GuiAutoScrollCtrl
- resetGhosting() : GameConnection
- resetState() : GuiButtonBaseCtrl
- resetTarget() : AITurretShape
- resize() : GuiControl
- resizeHeight : GuiWindowCtrl
- resizeWidth : GuiWindowCtrl
- resolveEffect : RenderFormatToken
- resolveGhostID() : NetConnection
- resolveObjectFromGhostIndex() : NetConnection
- respawn : TurretShape
- restitution : PhysicsShapeData, PxMaterial, WheeledVehicleTire, PhysicsDebrisData
- restorativeForce : HoverVehicleData
- restore() : PhysicsShape
- restoreWindow() : GuiCanvas
- resultArg : GFXSamplerStateData
- returnTab : GuiControlProfile
- reverb : SFXEnvironment
- REVERB_CORE0 : SFXEnvironment
- REVERB_CORE1 : SFXEnvironment
- REVERB_DECAYHFLIMIT : SFXEnvironment
- REVERB_DECAYTIMESCALE : SFXEnvironment
- REVERB_DIRECTHFAUTO : SFXDescription
- REVERB_ECHOTIMESCALE : SFXEnvironment
- REVERB_HIGHQUALITYDPL2REVERB : SFXEnvironment
- REVERB_HIGHQUALITYREVERB : SFXEnvironment
- REVERB_INSTANCE0 : SFXDescription
- REVERB_INSTANCE1 : SFXDescription
- REVERB_INSTANCE2 : SFXDescription
- REVERB_INSTANCE3 : SFXDescription
- REVERB_MODULATIONTIMESCALE : SFXEnvironment
- REVERB_REFLECTIONSDELYSCALE : SFXEnvironment
- REVERB_REFLECTIONSSCALE : SFXEnvironment
- REVERB_REVERBDELYSCALE : SFXEnvironment
- REVERB_REVERBScale : SFXEnvironment
- REVERB_ROOMAUTO : SFXDescription
- REVERB_ROOMHFAUTO : SFXDescription
- reverbAirAbsorptionFactor : SFXDescription
- reverbDelay : SFXEnvironment
• reverbDirect : SFXDescription
• reverbDirectHF : SFXDescription
• reverbDopplerFactor : SFXDescription
• reverbExclusion : SFXDescription
• reverbExclusionLFRatio : SFXDescription
• reverbFlags : SFXDescription
• reverbObstruction : SFXDescription
• reverbObstructionLFRatio : SFXDescription
• reverbOcclusion : SFXDescription
• reverbOcclusionDirectRatio : SFXDescription
• reverbOcclusionLFRatio : SFXDescription
• reverbOcclusionRoomRatio : SFXDescription
• reverbOutsideVolumeHF : SFXDescription
• reverbPan : SFXEnvironment
• reverbReverbRolloffFactor : SFXDescription
• reverbRoom : SFXDescription
• reverbRoomHF : SFXDescription
• reverbRoomRolloffFactor : SFXDescription
• reverseOrder : ParticleEmitterData
• reverseTextList : GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
• reverseThrustForce : HoverVehicleData
• rightPad : GuiGameListOptionsProfile
• rigidity : ForestItemData
• ringLifetime : SplashData
• rippleDir : WaterObject
• rippleMagnitude : WaterObject
• rippleSpeed : WaterObject
• rippleTex : WaterObject
• rippleTexScale : WaterObject
• rollForce : HoverVehicleData, FlyingVehicleData
• rolloffFactor : SFXDescription, SFXAmbience
• room : SFXEnvironment
• roomHF : SFXEnvironment
• roomLF : SFXEnvironment
• roomRolloffFactor : SFXEnvironment
• rotA : LightAnimData
• rotate : Item
• rotateWithCamVel : Precipitation
• rotation : ShapeBaseImageData, SceneObject
- `rotationalDrag`: `FlyingVehicleData`
- `rotationRange`: `ForestBrushElement`
- `rotKeys`: `LightAnimData`
- `rotPeriod`: `LightAnimData`
- `rotPivotOffset`: `Material`
- `rotSmooth`: `LightAnimData`
- `rotSpeed`: `Material`
- `rotZ`: `LightAnimData`
- `rowCount`: `GuiDynamicCtrlArrayControl, GuiTextListCtrl`
- `rows`: `GuiFrameSetCtrl`
- `rowSize`: `GuiDynamicCtrlArrayControl, GuiGameListMenuProfile, GuiControlArrayControl`
- `rowSpacing`: `GuiDynamicCtrlArrayControl, GuiControlArrayControl`
- `runEnergyDrain`: `PlayerData`
- `runForce`: `PlayerData`
- `runSurfaceAngle`: `PlayerData`
Here is a list of all class members with links to the classes they belong to:
- s -

- samplersDefined : GFXStateBlockData
  - samplerStates : GFXStateBlockData
  - samples : PxCloth
  - save() : SimObject, ActionMap, TerrainBlock
  - saveDataFile : Forest
  - saveFile() : SimXMLDocument
  - saveShape() : TSShapeConstructor
  - sbUsesNAColor : GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
  - scale : SceneObject, SplashData, ProjectileData
  - scaleExponent : ForestBrushElement
  - scaleMax : ForestBrushElement
  - scaleMin : ForestBrushElement
  - scaleU : GroundPlane
  - scaleV : GroundPlane
  - scanTickFrequency :AITurretShapeData
  - scanTickFrequencyVariance :AITurretShapeData
  - scatterDistance : SFXDescription, SFXEmitter
  - schedule() : SimObject
  - scopeToClient() : NetObject
  - screenError : TerrainBlock
  - screenToClient() : GuiCanvas
  - scriptAnimTransitionTime : ShapeBaseImageData
  - scrollDir : Material
  - scrollDirection : GuiAutoScrollCtrl
  - scrollOutOfSight : GuiAutoScrollCtrl
  - scrollSpeed : Material, GuiAutoScrollCtrl
  - scrollToBottom() : GuiMLTextCtrl, GuiScrollCtrl
  - scrollToObject() : GuiScrollCtrl
  - scrollToTag() : GuiMLTextCtrl
  - scrollToTop() : GuiMLTextCtrl, GuiScrollCtrl
  - scrollVisible() : GuiTreeViewCtrl, GuiTextListCtrl
  - scrollVisibleByObjectId() : GuiTreeViewCtrl
  - seed : fxFoliageReplicator, fxShapeReplicator, GroundCover
  - SegmentLength : River
  - select() : GuiTabPageCtrl
  - selectAllText() : GuiTextEditCtrl
  - selectedPage : GuiTabBookCtrl
- `selectItem()` : `GuiTreeViewCtrl`
- `selectOption()` : `GuiGameListOptionsCtrl`
- `selectPage()` : `GuiTabBookCtrl`
- `selectWindow()` : `GuiWindowCtrl`
- `selfCollision` : `PxCloth`
- `send()` : `RadialImpulseEvent`, `TCPObject`
- `separateAlphaBlendDefined` : `GFXStateBlockData`
- `separateAlphaBlendDest` : `GFXStateBlockData`
- `separateAlphaBlendEnable` : `GFXStateBlockData`
- `separateAlphaBlendOp` : `GFXStateBlockData`
- `separateAlphaBlendSrc` : `GFXStateBlockData`
- `seqNum` : `Marker`
- `sequence` : `TSShapeConstructor`
- `sequenceFramePerSec` : `Material`
- `sequenceSegmentSize` : `Material`
- `setActionThread()` : `Player`
- `setActive()` : `GuiControl`, `ParticleEmitterNode`
- `setAimLocation` : `AIClient`, `AIPlayer`
- `setAimObject()` : `AIPlayer`
- `setAlarmMode()` : `InteriorInstance`
- `setAllGunsFiring()` : `AITurretShape`
- `setAllHidden()` : `PxMultiActor`
- `setAllMeshesHidden()` : `ShapeBase`
- `setAllowManualFire()` : `TurretShape`
- `setAllowManualRotation()` : `TurretShape`
- `setAlpha()` : `GuiMLTextCtrl`
- `setAngularDrag()` : `Camera`
- `setAngularForce()` : `Camera`
- `setAngularVelocity()` : `Camera`
- `setArea()` : `MissionArea`
- `setArmThread()` : `Player`
- `setAttribute()` : `SimXMLDocument`
- `setAutoGenerated` : `Material`
- `setBatchSize()` : `River`
- `setBitmap()` : `GuiBitmapCtrl`, `GuiBitmapButtonCtrl`, `GuiIconButtonCtrl`, `GuiChunkedBitmapCtrl`, `GuiProgressBitmapCtrl`
- `setBlackOut()` : `GameConnection`
- `setBounds()` : `TSShapeConstructor`
- `setBrakeMultiplier()` : `Camera`
- `setBroken()` : `PxMultiActor`
- `setCameraFov()` : `ShapeBase`
- `setCameraObject()` : `GameConnection`
- `setCameraSpeed()` : `GuiObjectView`
- `setCanSave()` : `SimObject`
- `setCapability()` : `GFXCardProfilerAPI`
- `setCenter()` : `GuiControl`
- `setCheckmarkBitmapIndex()` : `GuiMenuBar`
- `setClassNamespace()` : `SimObject`
- `setCloaked()` : `ShapeBase`
- `setCollapsed()` : `GuiPaneControl`
- `setCollapseGroup()` : `GuiWindowCtrl`
- `setCollisionTimeout()` : `Item`
- `setColor()` : `GuiSwatchButtonCtrl`
- `setColumnOffset()` : `GuiFrameSetCtrl`
- `setCone()` : `SFXSource`
- `setConnectArgs()` : `GameConnection`
- `setContent()` : `GuiCanvas`
- `setControl()` : `GameBase`
- `setControlCameraFov()` : `GameConnection`
- `setControlObject()` : `GameConnection, Player`
- `setCurrent()` : `ArrayObject`
- `setCurrentLanguage()` : `LangTable`
- `setCurrentSlot()` : `SFXController`
- `setCurSel()` : `GuiListBoxCtrl`
- `setCurSelRange()` : `GuiListBoxCtrl`
- `setCursor()` : `GuiCanvas`
- `setCursorPos()` : `GuiTextEditCtrl, GuiCanvas`
- `setCursorPosition()` : `GuiMLTextCtrl`
- `setDamageFlash()` : `ShapeBase`
- `setDamageLevel()` : `ShapeBase`
- `setDamageState()` : `ShapeBase`
- `setDamageVector()` : `ShapeBase`
- `setDataBlock()` : `GameBase`
- `setDayLength()` : `TimeOfDay`
- `setDebug()` : `GuiTreeCtrl`
- `setDefaultLanguage()` : `LangTable`
- `setDetailLevel()` : `InteriorInstance`
- `setDetailLevelSize()` : `TSShapeConstructor`
- `setDrag()` : `Camera`
- `setEditOrbitMode()` : `Camera`
- `setEditOrbitPoint()` : `Camera`
- `setEditorOnly()` : `SimObject`
- `setEmitterDataBlock()` : `ParticleEmitterNode`
- `setEnergyLevel()` : `ShapeBase`
- `setEnumContent` : `GuiPopUpMenuCtrlEx`, `GuiPopUpMenuCtrl`
- `setExtent()` : `GuiControl`
- `setFadeTimes()` : `SFXSource`
- `setFieldType()` : `SimObject`
- `setFieldValue()` : `SimObject`
- `setFile()` : `GuiTheoraCtrl`
- `setFilename()` : `SimObject`
- `setFilter()` : `GuiDirectoryFileListCtrl`
- `setFilterText()` : `GuiTreeViewCtrl`
- `setFirstPerson()` : `GameConnection`
- `setFirstResponder()` : `GuiControl`
- `setFirstSelected()` : `GuiPopUpMenuCtrl`
- `setFlyForce()` : `Camera`
- `setFlyMode()` : `Camera`
- `setFocus()` : `GuiCanvas`
- `setFreeLook()` : `AIConnection`
- `setGraphType()` : `GuiGraphCtrl`
- `setGunSlotFiring()` : `AITurretShape`
- `setHidden()` : `SimObject`, `ShapeBase`
- `setImageAltTrigger()` : `ShapeBase`
- `setImageAmmo()` : `ShapeBase`
- `setImageGenericTrigger()` : `ShapeBase`
- `setImageLoaded()` : `ShapeBase`
- `setImageScriptAnimPrefix()` : `ShapeBase`
- `setImageTarget()` : `ShapeBase`
- `setImageTrigger()` : `ShapeBase`
- `setInternalName()` : `SimObject`
- `setInvincibleMode()` : `ShapeBase`
- `setIsExpanded()` : `SimObject`
- `setIsSelected()` : `SimObject`
- `setItemColor()` : `GuiListBoxCtrl`
- `setItemImages()` : `GuiTreeViewCtrl`
- `setItemText()` : `GuiListBoxCtrl`
- `setItemTooltip()` : `GuiListBoxCtrl`, `GuiTreeViewCtrl`
- `setJoinPassword()` : `GameConnection`
- `setKey()` : `ArrayObject`
- `setLagIcon()` : `GameConnection`
- `setLastTTL()` : `DebugDrawer`
- `setLastZTest()` : `DebugDrawer`
- `setLightAmbient()` : `GuiObjectView`
- `setLightColor()` : `GuiObjectView`
- `setLightDirection()` : `GuiObjectView`
- `setLightEnabled()` : `LightBase`
- `setLocked()` : `SimObject`
- `setMass()` : `Camera`
- `setMaxDivisionSize()` : `River`
- `setMenuBitmapIndex()` : `GuiMenuBar`
- `setMenuItemBitmap()` : `GuiMenuBar`
- `setMenuItemChecked()` : `GuiMenuBar`
- `setMenuItemEnable()` : `GuiMenuBar`
- `setMenuItemSubmenuState()` : `GuiMenuBar`
- `setMenuItemText()` : `GuiMenuBar`
- `setMenuItemVisible()` : `GuiMenuBar`
- `setMenuMargins()` : `GuiMenuBar`
- `setMenuText()` : `GuiMenuBar`
- `setMenuVisible()` : `GuiMenuBar`
- `setMeshHidden()` : `ShapeBase`, `PxMultiActor`
- `setMeshMaterial()` : `TSShapeConstructor`
- `setMeshSize()` : `TSShapeConstructor`
- `setMeshType()` : `TSShapeConstructor`
- `setMessage()` : `SimpleNetObject`
- `setMetersPerSegment()` : `River`
- `setMissionCRC()` : `GameConnection`
- `setModel()` : `GuiObjectView`
- `setMount()` : `GuiObjectView`
- `setMountedModel()` : `GuiObjectView`
- `setMountSkin()` : `GuiObjectView`
- `setMove()` : `AIConnection`
- `setMoveDestination()` : `AIClient`, `AIPlayer`
- `setMoveSpeed()` : `AIClient`, `AIPlayer`
- `setMultipleSelection()` : `GuiListBoxCtrl`
- setName() : SimObject
- setNameChangeAllowed() : SimObject
- setNewtonFlyMode() : Camera
- setNodeDepth() : MeshRoad, River
- setNodeParent() : TSShapeConstructor
- setNodeTransform() : TSShapeConstructor
- setNoneSelected : GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
- setObjectAttributes() : SimXMLDocument
- setObjectNode() : TSShapeConstructor
- setOffset() : Camera
- setOptions() : GuiGameListOptionsCtrl
- setOrbitDistance() : GuiObjectView
- setOrbitMode() : Camera
- setOrbitObject() : Camera
- setOrbitPoint() : Camera
- setPath() : GuiDirectoryFileListCtrl
- setPercentage() : Precipitation
- setPitch() : SFXSource
- setPlay() : TimeOfDay
- setPosition() : GuiControl, PathCamera, StreamObject, SFXSound
- setPositionGlobal() : GuiControl
- setProcessTicks() : GuiTickCtrl
- setProfile() : GuiControl
- setRechargeRate() : ShapeBase
- setRepairRate() : ShapeBase
- setReverseTime() : GuiClockHud
- setRotation() : Camera
- setRowActive() : GuiTextListCtrl
- setRowById() : GuiTextListCtrl
- setRowEnabled() : GuiGameListMenuCtrl
- setRowLabel() : GuiGameListMenuCtrl
- setRowOffset() : GuiFrameSetCtrl
- setScale() : SceneObject
- setScopeAlways() : NetObject
- setScrollPosition() : GuiScrollCtrl
- setSelected() : GuiListBoxCtrl, GuiGameListMenuCtrl, GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
- setSelectedById() : GuiTextListCtrl
- `setSelectedRow()` : `GuiTextListCtrl`
- `setSeq()` : `GuiObjectView`
- `setSequenceBlend()` : `TSShapeConstructor`
- `setSequenceCyclic()` : `TSShapeConstructor`
- `setSequenceGroundSpeed()` : `TSShapeConstructor`
- `setSequencePriority()` : `TSShapeConstructor`
- `setShaderConst()` : `PostEffect`
- `setShaderConsts()` : `PostEffect`
- `setShaderMacro()` : `PostEffect`
- `setName()` : `ShapeBase`
- `setSimulatedNetParams()` : `NetConnection`
- `setSkin()` : `GuiObjectView`
- `setSkinName()` : `ShapeBase`
- `setSpeedMultiplier()` : `Camera`
- `setState()` : `PathCamera`
- `setStateOn()` : `GuiButtonBaseCtrl`, `GuiCheckBoxCtrl`
- `setSubmenuItemChecked()` : `GuiMenuBar`
- `setSuperClassNamespace()` : `SimObject`
- `setTarget()` : `PathCamera`
- `setTargetObject()` : `AIClient`
- `setText()` : `GuiButtonBaseCtrl`, `GuiTextCtrl`, `GuiTextEditCtrl`, `GuiMLTextCtrl`, `GuiPopUpMenuCtrlEx`
- `setTextID()` : `GuiButtonBaseCtrl`, `GuiTextCtrl`
- `setTexture()` : `PostEffect`
- `setThreadDir()` : `ShapeBase`
- `setThreadPosition()` : `ShapeBase`
- `setThreadTimeScale()` : `ShapeBase`
- `setTime()` : `GuiClockHud`
- `setTimeOfDay()` : `TimeOfDay`
- `setTrackObject()` : `Camera`
- `setTransform()` : `SceneObject`, `SFXSource`
- `setTrigger()` : `AIConnection`
- `setTurbulence()` : `Precipitation`
- `setTurretEulerRotation()` : `TurretShape`
- `setTurretState()` : `AITurretShape`
- `setValidEditOrbitPoint()` : `Camera`
- `setValue()` : `GuiControl`, `GuiBitmapCtrl`, `ArrayObject`, `GuiSliderCtrl`
- `setVelocity()` : `ShapeBase`, `Camera`
- `setVideoMode(): GuiCanvas`
- `setVisible(): GuiControl`
- `setVolume(): SFXSource`
- `setWeaponLeadVelocity(): ALTurretShape`
- `setWheelPowered(): WheeledVehicle`
- `setWheelSpring(): WheeledVehicle`
- `setWheelSteering(): WheeledVehicle`
- `setWheelTire(): WheeledVehicle`
- `setWhiteOut(): ShapeBase`
- `setWindowPosition(): GuiCanvas`
- `setWindowTitle(): GuiCanvas`
- `shader: CustomMaterial, PostEffect`
- `shadowDarkenColor: LightBase, LightDescription, ScatterSky, Sun`
- `shadowDistance: LightBase, LightDescription, ScatterSky, Sun`
- `shadowEnable: ShapeBaseData`
- `shadowMaxVisibleDistance: ShapeBaseData`
- `shadowProjectionDistance: ShapeBaseData`
- `shadowSize: ShapeBaseData`
- `shadowSoftness: LightBase, LightDescription, ScatterSky, Sun`
- `shadowSphereAdjust: ShapeBaseData`
- `shadowType: LightBase, LightDescription, ScatterSky, Sun`
- `shakeCamera: ExplosionData, ShapeBaseImageData`
- `ShapeCount: fxShapeReplicator`
- `shapeCullRadius: GroundCover`
- `shapeFile: PhysicsDebrisData, WheeledVehicleTire, ShapeBaseData, DebrisData, RenderShapeExample, fxShapeReplicator, ForestItemData, GuiObjectView, ShapeBaseImageData`
- `shapeFileFP: ShapeBaseImageData`
- `shapeFilename: GroundCover`
- `shapeName: PhysicsShapeData, fxShapeReplicatedStatic, TSStatic, PxMultiActorData`
- `shapeNameFP: PlayerData`
- `ShapeRetries: fxShapeReplicator`
- `ShapeRotateMax: fxShapeReplicator`
- `ShapeRotateMin: fxShapeReplicator`
- ShapeScaleMax: `fxShapeReplicator`
- ShapeScaleMin: `fxShapeReplicator`
- shapesCastShadows: `GroundCover`
- shellExitDir: `ShapeBaseImageData`
- shellExitVariance: `ShapeBaseImageData`
- shellVelocity: `ShapeBaseImageData`
- show(): `PfxVis`
- showBatches: `DecalRoad, MeshRoad`
- showClassNameForUnnamedObjects: `GuiTreeViewCtrl`
- showClassNames: `GuiTreeViewCtrl`
- showCursor(): `GuiCanvas`
- showDust: `Material`
- showEnergy: `GuiHealthTextHud`
- showFill: `GuiClockHud, GuiHealthBarHud, GuiHealthTextHud, GuiShapeNameHud`
- showFootprints: `Material`
- showFrame: `GuiShapeNameHud, GuiHealthBarHud, GuiHealthTextHud, GuiClockHud`
- showInternalNames: `GuiTreeViewCtrl`
- showItemRenameCtrl(): `GuiTreeViewCtrl`
- showNodes: `River`
- showObjectIds: `GuiTreeViewCtrl`
- showObjectNames: `GuiTreeViewCtrl`
- ShowPlacementArea: `fxFoliageReplicator, fxShapeReplicator`
- showRiver: `River`
- showRoad: `DecalRoad, MeshRoad`
- showRoot: `GuiTreeViewCtrl`
- showSpline: `DecalRoad, MeshRoad, River`
- showTrueValue: `GuiHealthTextHud`
- showWalls: `River`
- showWireframe: `River`
- sideMaterial: `MeshRoad`
- simpleServerCollision: `ItemData`
- simType: `PhysicsShapeData`
- singleDetailSize: `TSShapeConstructor`
- singlePlayerOnly: `PxMultiActorData`
- sinkAllKeyEvents: `GuiTextEditCtrl`
- sinkMax: `ForestBrushElement`
- sinkMin: `ForestBrushElement`
- `sinkRadius`: `ForestBrushElement`
- `size`: `PxCloth`, `DecalData`, `GuiPopUpMenuCtrl`, `GuiPopUpMenuCtrlEx`
- `sizeExponent`: `GroundCover`
- `sizeIconToButton`: `GuiIconButtonCtrl`
- `sizeMax`: `GroundCover`
- `sizeMin`: `GroundCover`
- `sizes`: `ExplosionData`, `ParticleData`
- `sizeToContents()`: `GuiRolloutCtrl`
- `skin`: `ShapeBase`, `fxShapeReplicatedStatic`, `GuiObjectView`, `TSStatic`
- `skip`: `PostEffect`
- `skyBrightness`: `ScatterSky`
- `slopeMax`: `ForestBrushElement`
- `slopeMin`: `ForestBrushElement`
- `smoothingType`: `Marker`
- `snap`: `GuiSliderCtrl`
- `snapOnMaxBounce`: `DebrisData`
- `softImpactSound`: `RigidShapeData`, `VehicleData`
- `softImpactSpeed`: `VehicleData`, `RigidShapeData`
- `softnessDistance`: `ParticleEmitterData`
- `softSplashSoundVelocity`: `RigidShapeData`, `VehicleData`
- `sort`: `GuiPopUpMenuCtrl`, `SimSet`, `ArrayObject`, `GuiPopUpMenuCtrlEx`, `GuiTextListCtrl`, `GuiTreeViewCtrl`
- `sorta()`: `ArrayObject`
- `sortd()`: `ArrayObject`
- `sortf()`: `ArrayObject`
- `sortfd()`: `ArrayObject`
- `sortfk()`: `ArrayObject`
- `sortfkd()`: `ArrayObject`
- `sortID()`: `GuiPopUpMenuCtrlEx`, `GuiPopUpMenuCtrl`
- `sortk()`: `ArrayObject`
- `sortka()`: `ArrayObject`
- `sortkd()`: `ArrayObject`
- `sortn()`: `ArrayObject`
- `sortna()`: `ArrayObject`
- `sortnd()`: `ArrayObject`
- `sortnk()`: `ArrayObject`
- `sortnka()`: `ArrayObject`
• sortnkd() : ArrayObject
• sortNumerical() : GuiTextListCtrl
• sortParticles : ParticleEmitterData
• sound : ProjectileData
• soundAmbience : LevelInfo, SFXSpace, WaterObject, Zone
• soundButtonDown : GuiControlProfile
• soundButtonOver : GuiControlProfile
• soundDistanceModel : LevelInfo
• soundProfile : ExplosionData, SplashData, PrecipitationData
• soundTrack : SFXAmbience
• sourceGroup : SFXDescription, SFXEmitter
• sourceObject : Projectile
• sourceSlot : Projectile
• spawnClass : SpawnSphere
• spawnDatablock : SpawnSphere
• spawnObject() : SpawnSphere
• spawnProperties : SpawnSphere
• spawnScript : SpawnSphere
• spawnTransform : SpawnSphere
• specular : Material
• specularColor : WaterObject
• specularMap : Material
• specularPower : WaterObject, Material
• speedMultiplier : Camera
• sphereWeight : SpawnSphere
• spinRandomMax : ParticleData
• spinRandomMin : ParticleData
• spinSpeed : ParticleData
• Splash : PlayerData, ProjectileData
• splashAngle : PlayerData
• splashEmitter : RigidShapeData, VehicleData, PlayerData
• splashesPerSide : PrecipitationData
• splashFreqMod : RigidShapeData, PlayerData, VehicleData
• splashMS : Precipitation
• splashShader : PrecipitationData
• splashSize : Precipitation
• splashTexture : PrecipitationData
• splashVelEpsilon : VehicleData, RigidShapeData, PlayerData
• splashVelocity : PlayerData
- splitPoint : GuiSplitContainer
- splitterSize : GuiSplitContainer
- sprintCanJump : PlayerData
- sprintEnergyDrain : PlayerData
- sprintForce : PlayerData
- sprintPitchScale : PlayerData
- sprintStrafeScale : PlayerData
- sprintTrigger : Player
- sprintYawScale : PlayerData
- squareSize : GroundPlane, TerrainBlock
- squealSound : WheeledVehicleData
- stabDampingConstant : HoverVehicleData
- stabLenMax : HoverVehicleData
- stabLenMin : HoverVehicleData
- stabSpringConstant : HoverVehicleData
- stackingType : GuiStackControl
- startDelay : GuiAutoScrollCtrl
- startDragging() : GuiDragAndDropControl
- startFade() : ShapeBase
- startLoaded : TurretShapeData
- startRadius : SplashData
- startRecording() : GameConnection
- startScanForTargets() : AITurretShape
- startTime : TimeOfDay
- startTrackingTarget() : AITurretShape
- state : SFXPlayList
- stateAllowImageChange : ShapeBaseImageData
- stateAlternateFire : ShapeBaseImageData
- stateBlock : CustomMaterial, PostEffect
- stateDirection : AITurretShapeData, ShapeBaseImageData
- stateEjectShell : ShapeBaseImageData
- stateEmitter : ShapeBaseImageData
- stateEmitterNode : ShapeBaseImageData
- stateEmitterTime : ShapeBaseImageData
- stateEnergyDrain : ShapeBaseImageData
- stateFire : AITurretShapeData, ShapeBaseImageData
- stateIgnoreLoadedForReady : ShapeBaseImageData
- stateLoadedFlag : ShapeBaseImageData
- stateMode : SFXPlayList
- `stateName`: `AITurretShapeData, ShapeBaseImageData`
- `stateRecoil`: `ShapeBaseImageData`
- `stateReload`: `ShapeBaseImageData`
- `states`: `SFXAmbience`
- `stateScaleAnimation`: `ShapeBaseImageData, AITurretShapeData`
- `stateScaleAnimationFP`: `ShapeBaseImageData`
- `stateScaleShapeSequence`: `ShapeBaseImageData`
- `stateScan`: `AITurretShapeData`
- `stateScript`: `AITurretShapeData, ShapeBaseImageData`
- `stateSequence`: `AITurretShapeData, ShapeBaseImageData`
- `stateSequenceNeverTransition`: `ShapeBaseImageData`
- `stateSequenceRandomFlash`: `ShapeBaseImageData`
- `stateSequenceTransitionIn`: `ShapeBaseImageData`
- `stateSequenceTransitionOut`: `ShapeBaseImageData`
- `stateSequenceTransitionTime`: `ShapeBaseImageData`
- `stateShapeSequence`: `ShapeBaseImageData`
- `stateSound`: `ShapeBaseImageData`
- `stateSpinThread`: `ShapeBaseImageData`
- `stateTimeoutValue`: `AITurretShapeData, ShapeBaseImageData`
- `stateToken`: `RenderPassStateBin`
- `stateTransitionGeneric0In`: `ShapeBaseImageData`
- `stateTransitionGeneric0Out`: `ShapeBaseImageData`
- `stateTransitionGeneric1In`: `ShapeBaseImageData`
- `stateTransitionGeneric1Out`: `ShapeBaseImageData`
- `stateTransitionGeneric2In`: `ShapeBaseImageData`
- `stateTransitionGeneric2Out`: `ShapeBaseImageData`
- `stateTransitionGeneric3In`: `ShapeBaseImageData`
- `stateTransitionGeneric3Out`: `ShapeBaseImageData`
- `stateTransitionOnActivated`: `AITurretShapeData`
- `stateTransitionOnAltTriggerDown`: `ShapeBaseImageData`
- `stateTransitionOnAltTriggerUp`: `ShapeBaseImageData`
- `stateTransitionOnAmmo`: `ShapeBaseImageData`
- `stateTransitionOnAtRest`: `AITurretShapeData`
- `stateTransitionOnDeactivated`: `AITurretShapeData`
- `stateTransitionOnLoaded`: `ShapeBaseImageData`
- `stateTransitionOnMotion`: `ShapeBaseImageData`
- `stateTransitionOnNoAmmo`: `ShapeBaseImageData`
- stateTransitionOnNoMotion: ShapeBaseImageData
- stateTransitionOnNoTarget: AITurretShapeData, ShapeBaseImageData
- stateTransitionOnNotAtRest: AITurretShapeData
- stateTransitionOnNotLoaded: ShapeBaseImageData
- stateTransitionOnNotWet: ShapeBaseImageData
- stateTransitionOnTarget: ShapeBaseImageData, AITurretShapeData
- stateTransitionOnTimeout: ShapeBaseImageData, AITurretShapeData
- stateTransitionOnTriggerDown: ShapeBaseImageData
- stateTransitionOnTriggerUp: ShapeBaseImageData
- stateTransitionOnWet: ShapeBaseImageData
- stateWaitForTimeout: ShapeBaseImageData, AITurretShapeData
- static: Item
- staticFriction: PhysicsShapeData, PhysicsDebrisData, WheeledVehicleTire, PxMaterial
- staticOnMaxBounce: DebrisData
- statusCallback: SFXSource
- steeringForce: HoverVehicleData, FlyingVehicleData
- steeringRollForce: FlyingVehicleData
- stencilDefined: GFXStateBlockData
- stencilEnable: GFXStateBlockData
- stencilFailOp: GFXStateBlockData
- stencilFunc: GFXStateBlockData
- stencilMask: GFXStateBlockData
- stencilPassOp: GFXStateBlockData
- stencilRef: GFXStateBlockData
- stencilWriteMask: GFXStateBlockData
- stencilZFailOp: GFXStateBlockData
- sticky: ItemData
- stop(): AIPlayer, SFXEmitter, SFXSource, AIClient, GuiTheoraCtrl
- stopAudio(): ShapeBase
- stopOnSleep: GuiTheoraCtrl
- stopRecording(): GameConnection
- stopScanForTargets(): AITurretShape
- stopThread(): ShapeBase
• stopTrackingTarget() : AITurretShape
• strafeThrustForce : HoverVehicleData
• streamPacketSize : SFXDescription
• streamReadAhead : SFXDescription
• strength : ForestWindEmitter
• strikeObject() : Lightning
• strikeRadius : Lightning
• strikeRandomPoint() : Lightning
• strikeSound : LightningData
• strikesPerMinute : Lightning
• strikeTextures : LightningData
• strikeWidth : Lightning
• string : PxMultiActorData
• SubdivideLength : River
• subExplosion : ExplosionData
• subscribe() : EventManager
• subSurface : Material
• subSurfaceColor : Material
• subSurfaceRolloff : Material
• sunScale : ScatterSky
• superClass : SimObject
• surface : ConvexShape
• SwayMagFront : fxFoliageReplicator
• SwayMagSide : fxFoliageReplicator
• SwayOn : fxFoliageReplicator
• SwaySync : fxFoliageReplicator
• swimBoundingBox : PlayerData
• swimForce : PlayerData
Here is a list of all class members with links to the classes they belong to:
- t -

- tab : GuiControlProfile
- tabComplete : GuiTextEditCtrl
- tabHeight : GuiTabBookCtrl
- tabMargin : GuiTabBookCtrl
- tabPosition : GuiTabBookCtrl
- tabSize : GuiTreeViewCtrl
- tail : GuiSpeedometerHud
- target : CustomMaterial, PostEffect
- targetClear : PostEffect
- targetClearColor : PostEffect
- targetDepthStencil : PostEffect
- targetFormat : PostEffect
- targetScale : PostEffect
- targetSize : PostEffect
- team : WayPoint
- terminalVelocity : DebrisData
- TerrainAlignment : fxShapeReplicator
- terrainFile : TerrainBlock
- texCols : DecalData
- texDirection : BasicClouds, CloudLayer
- texFactor : SplashData
- texOffset : BasicClouds
- texRows : DecalData
- texScale : BasicClouds, CloudLayer
- texSize : Sun, LightBase, ReflectorDesc, LightDescription, ScatterSky
- texSpeed : BasicClouds, CloudLayer
- text : GuiButtonBaseCtrl, GuiWindowCtrl, GuiTextCtrl, GuiMLTextCtrl
- textColor : GuiHealthTextHud, GuiShapeNameHud, GuiClockHud
- textID : GuiButtonBaseCtrl, GuiTextCtrl
- textLocation : GuiIconButtonCtrl
- textMargin : GuiIconButtonCtrl
- textOffset : GuiControlProfile, GuiTreeViewCtrl
- texture : DebrisData, SplashData, BasicClouds, CloudLayer, PostEffect
• textureColorOp : GFXSamplerStateData
• textureCoordCount : DecalData
• textureCoords : DecalData, ParticleData
• textureFactor : GFXStateBlockData
• textureLength : DecalRoad, MeshRoad
• textureName : ParticleData, ParticleEmitterData
• textureTransform : GFXSamplerStateData
• texWrap : SplashData
• theoraFile : GuiTheoraCtrl
• thetaMax : ParticleEmitterData
• thetaMin : ParticleEmitterData
• thickness : PxCloth
• thunderSounds : LightningData
• tickCommand : Trigger
• tickPeriodMS : TriggerData
• ticks : GuiSliderCtrl
• tightnessCoefficient : ForestItemData
• tile : GuiChunkedBitmapCtrl
• time : TimeOfDay
• timeMultiple : ParticleEmitterNodeData
• times : ExplosionData, SplashData, ParticleData
• tireEmitter : WheeledVehicleData
• title : FileDialog
• toggle() : RenderPassStateToken, PostEffect
• toggleCollapse() : GuiRolloutCtrl
• toggleCollapseGroup() : GuiWindowCtrl
• toggleDrawing() : DebugDrawer
• toggleExpanded() : GuiRolloutCtrl
• toggleFreeze() : DebugDrawer
• toggleFullscreen() : GuiCanvas
• toggleHideSelection() : GuiTreeViewCtrl
• toggleLockSelection() : GuiTreeViewCtrl
• toneMap : Material
• tooltip : GuiControl
• tooltipOnWidthOnly : GuiTreeViewCtrl
• tooltipProfile : GuiControl
• topMaterial : MeshRoad
• toQueue : MessageForwarder
• trace : SFXController, SFXPlayList
- track : SFXEmitter, SFXPlayList
- trackLostTargetTime : AITurretShapeData
- trailEmitter : FlyingVehicleData
- transcoder : GuiTheoraCtrl
- transitionIn : SFXPlayList
- transitionOut : SFXPlayList
- transitionToLand : PlayerData
- translucent : Material
- translucentBlendOp : Material
- translucentZWrite : Material
- transmitDataBlocks() : GameConnection
- transmitPaths() : NetConnection
- triangleCollision : PxCloth
- triggerDelay : ProximityMineData
- triggerDustHeight : VehicleData, RigidShapeData
- triggerOnOwner : ProximityMineData
- triggerRadius : ProximityMineData
- triggerSound : ProximityMineData
- triggerSpeed : ProximityMineData
- triggerTrailHeight : HoverVehicleData
- trunkBendScale : ForestItemData
- turboFactor : HoverVehicleData
- turbulenceFrequency : ForestWindEmitter
- turbulenceSpeed : Precipitation
- turbulenceStrength : ForestWindEmitter
- type : Marker, GuiSeparatorCtrl
Here is a list of all class members with links to the classes they belong to:
unbind() : ActionMap
unbindObj() : ActionMap
underwaterColor : WaterObject
underwaterExplosion : ShapeBaseData
uniqueKey() : ArrayObject
uniqueValue() : ArrayObject
unit : TSShapeConstructor
unmount() : SceneObject
unmountImage() : ShapeBase
unmountObject() : SceneObject
unproject() : GuiTSCtrl
unregisterEvent() : EventManager
upAxis : TSShapeConstructor
updateDelay : DecalRoad
updateFaces() : CubemapData
updateSizes() : GuiFrameSetCtrl
updateStack() : GuiStackControl
upMaxSpeed : PlayerData
upResistFactor : PlayerData
upResistSpeed : PlayerData
useAmbientLightColor : Zone
useAnisotropic : Material
useCreateHeight() : FlyingVehicle
UseCulling : fxFoliageReplicator
useCustomReverb : SFXDescription
UseDebugInfo : fxFoliageReplicator
useDevicePixVersion : ShaderData
useEmitterColors : ParticleEmitterData
useEmitterSizes : ParticleEmitterData
useEyeNode : ShapeBaseImageData
useEyePoint : ShapeBaseData
useFog : Lightning
useHardware : SFXDescription
useInspectorTooltips : GuiTreeViewCtrl
useInvAlpha : ParticleData
useLighting : Precipitation
useModifiers : GuiBitmapButtonCtrl
- useMouseEvents: GuiButtonBaseCtrl
- useNightCubemap: ScatterSky
- useOcclusionQuery: WaterObject, ReflectorDesc
- useRadiusMass: DebrisData
- useRemainderDT: ShapeBaseImageData
- usesEnergy: ShapeBaseImageData
- useSiblingScroller: GuiConsoleEditCtrl
- useSideProjection: TerrainMaterial
- useStates: GuiBitmapButtonCtrl
- useTrackDescriptionOnly: SFXEmitter
- useTrueBillboards: Precipitation, fxFoliageReplicator
- useTurbulence: Precipitation
- useURLMouseCursor: GuiMLTextCtrl
- useVariable: GuiChunkedBitmapCtrl
- useWind: Precipitation
Here is a list of all class members with links to the classes they belong to:
- v -

- validate : GuiTextEditCtrl
- validateCameraFov() : ShapeBase
- value : SFXParameter, GuiSliderCtrl
- variable : GuiControl
- vehicleDismountTrigger : Player
- velInheritFactor : ProjectileData
- velocity : DebrisData, SplashData, ParticleEmitterNode
- velocityMod : PhysicalZone
- velocityVariance : DebrisData, ParticleEmitterData
- version : CustomMaterial
- vertColor : Material
- vertexColorEnable : GFXStateBlockData
- vertFactor : RigidShapeData, HoverVehicleData
- verticalOffset : GuiShapeNameHud
- verticalSurfaceForce : FlyingVehicleData
- vertLit : Material
- vertSizing : GuiControl
- vertStacking : GuiStackControl
- vertThrustMultiple : FlyingVehicleData
- ViewClosest : fxFoliageReplicator
- ViewDistance : fxFoliageReplicator
- viscosity : WaterObject
- visible : GuiControl
- visibleDistance : LevelInfo
- volume : SFXDescription, SFXEmitter
- volumeScale : SFXPlayList
- volumeScaleVariance : SFXPlayList
- vScrollBar : GuiScrollCtrl
Here is a list of all class members with links to the classes they belong to:
- w -

- waitTime : GuiFadeinBitmapCtrl
  - warningColor : GuiHealthTextHud
  - warningFlashes() : Lightning
  - warnThreshold : GuiHealthTextHud
  - waterBreathSound : PlayerData
  - waterDampingScale : PhysicsShapeData, PhysicsDebrisData
  - waterDragScale : PxMultiActorData
  - waterExplosion : ProjectileData
  - waterFogDensity : WaterObject
  - waterFogDensityOffset : WaterObject
  - waterWakeSound : RigidShapeData, VehicleData
  - waveAmp : Material
  - waveDir : WaterObject
  - waveFreq : Material
  - waveMagnitude : WaterObject
  - waveSpeed : WaterObject
  - waveType : Material
  - weaponLeadVelocity : AITurretShapeData
  - weaponLinkType : TurretShapeData
  - wetDarkening : WaterObject
  - wetDepth : WaterObject
  - WheelImpactSound : WheeledVehicleData
  - width : SplashData, GuiSpeedometerHud
  - widthSubdivisions : MeshRoad
  - willFirstRespond : GuiScrollCtrl
  - windCoefficient : ParticleData
  - windDirection : GroundCover
  - windEnabled : ForestWindEmitter
  - windGustFrequency : GroundCover
  - windGustLength : GroundCover
  - windGustStrength : GroundCover
  - windScale : ForestItemData, GroundCover
  - windSpeed : CloudLayer
  - windTurbulenceFrequency : GroundCover
  - windTurbulenceStrength : GroundCover
  - wireframe : MeshRoad, WaterObject, DecalRoad
  - workingQueryBoxSizeMultiplier : Vehicle
• workingQueryBoxStaleThreshold : Vehicle
• wrap : GuiBitmapCtrl
• writeChangeSet() : TSShapeConstructor
• writeLine() : StreamObject, FileObject
• writeLongString() : StreamObject
• writeObject() : FileObject
• writeString() : StreamObject
Here is a list of all class members with links to the classes they belong to:
- z -

- zBias : GFXStateBlockData
- zDefined : GFXStateBlockData
- zEnable : GFXStateBlockData
- zFunc : GFXStateBlockData
- zOffset : GroundCover
- zoneGroup : InteriorInstance, Zone
- zRotOnly : TurretShapeData
- zSlopeBias : GFXStateBlockData
- zWriteEnable : GFXStateBlockData
- a -

- activePlugins : $BasicLightManagerStats
- alwaysUseDebugOutput : $Con
- ambientUpdateTime : $SFX
- autoDeleteTicks : $ProxMine
- b -

- Back : $Ease
- backgroundSleepTime : $platform
- Bounce : $Ease

- c -

- cameraFOV : $pref::WorldEditor
- canShadow : $pref::imposter
- Circular : $Ease
- CloakSpeed : $SB
- Cubic : $Ease

- d -

- debugRender : $Decals
- defaultAnisotropy : $pref::Video
- densityScale : $pref::GroundCover
- detailAdjust : $pref::TS , $pref::Interior
- detailAdjustScale : $pref::PSSM
- detailScale : $pref::Terrain
- DEVICE_CAPS_DSPEFFECTS : $SFX
- DEVICE_CAPS_FMODDESIGNER : $SFX
- DEVICE_CAPS_MULTILISTENER : $SFX
- DEVICE_CAPS_OCCLUSION : $SFX
- DEVICE_CAPS_REVERB : $SFX
- DEVICE_CAPS_VOICEMANAGEMENT : $SFX
- DEVICE_INFO_CAPS : $SFX
- DEVICE_INFO_MAXBUFFERS : $SFX
- DEVICE_INFO_NAME : $SFX
- DEVICE_INFO_PROVIDER : $SFX
- DEVICE_INFO_USEHARDWARE : $SFX
- DFDec : $SB
- disable : $pref::Shadows , $Shadows
- disableOcclusionQuery : $gfx
- disableSoftware : $pref::SFX::FMOD
- disableTerrainOcclusion : $Scene
- disableTrueReflections : $pref::Water
- disableVerticalSync : $pref::Video
- disableZoneCulling : $Scene
- disassembleAllShaders : $gfx
- distanceScale : $pref::Camera
- DontRestrictOutside : $Interior
- DSoundHRTF : $pref::SFX::FMOD

- elapsedUpdateMs : $BasicLightManagerStats
- Elastic : $Ease
- enableBadWordFilter : $pref
- enabled : $pref::Decals
- enablePostEffects : $pref
- enableProfile : $pref::SFX::FMOD
- Exponential : $Ease

- fadeEndPixelSize : $pref::ProjectedShadow
- fadeStartPixelSize : $pref::ProjectedShadow
- File : $Con
- filterMode : $pref::shadows
- fmodCoreMem : $SFX::Device
- fmodEventMem : $SFX::Device
- fmodNumEventSources : $SFX::Device
- forcedPixVersion : $pref::Video
- frameCount : $TSControl
- frameLimitMS : $pref::Reflect
- frequency : $Sampler
- FullCorrectionDistance : $SB

- In : $Ease
- InOut : $Ease
- isSinglePlayer : $Physics

- j -
- l -
  - JoystickEnabled : $pref::Input

- m -
  - LagThreshold : $pref::Net
  - lifetimeScale : $pref::PhysicsDebris
  - lifeTimeScale : $pref::Decals
  - Linear : $Ease
  - lockCull : $Scene
  - lodScale : $pref::Terrain
  - logBufferEnabled : $Con

- n -
  - maxInstancingVerts : $pref::TS
  - maxOccludersPerZone : $Scene

- o -
  - objectCopyFailures : $Con
  - occluderMinHeightPercentage : $Scene
  - occluderMinWidthPercentage : $Scene
  - Out : $Ease

- p -
  - PacketRateToClient : $pref::Net
  - PacketRateToServer : $pref::Net
  - PacketSize : $pref::Net
- p -
  - parameterUpdateTime : $SFX
  - pluginPath : $pref::SFX::FMOD
  - poolBuffers : $Decals
  - printLevel : $Con
  - PSSMDebugRender : $AL

- q -
  - Quadratic : $Ease
  - Quartic : $Ease
  - Quintic : $Ease

- r -
  - refractTexScale : $pref::Reflect
  - renderBoundingBoxes : $Scene
  - renderLightFrustums : $Light
  - renderViz : $Light
  - Root : $Con

- s -
  - shadowFilterDistance : $BasicLightManager
  - shadowsUpdated : $BasicLightManagerStats
  - Sinusoidal : $Ease
  - skipLoadDLs : $pref::TS
  - skipRenderDLs : $pref::TS
  - smallestVisiblePageSize : $pref::PSSM , $pref::TS
  - sourceUpdateTime : $SFX
  - sphereDistanceTolerance : $Decals
  - sphereRadiusTolerance : $Decals

- t -
  - textureReductionLevel : $pref::Video
  - textureScalar : $pref::Shadows
  - threadCount : $pref::Physics
  - timeManagerProcessInterval : $platform

- u -
- v -
  - VertexLighting : $pref::Interior
  - visibleDistanceScale : $pref::WorldEditor

- w -
  - warnUndefinedVariables : $Con
  - windEffectRadius : $pref
  - wireframe : $gfx
  - WODec : $SB
- b -

- bind() : ActionMap
- bindCmd() : ActionMap
- bindObj() : ActionMap
• bringToFront() : SimSet
• buildIconTable() : GuiTreeViewCtrl
- c -

- calculateViewDistance() : GuiTSCtrl
- call() : SimObject
- callOnChildren() : SimSet
- callOnChildrenNoRecurse() : SimSet
- canCloak() : ShapeBase
- canRenameObject() : GuiTreeViewCtrl
- changeMaterial() : TSStatic, InteriorInstance, ShapeBase
- changeTextById() : GuiPopUpMenuCtrl
- chaseCam() : GameConnection
- checkDeployPos() : ShapeBaseData
- checkDismountPoint() : Player
- checkMaxRate() : NetConnection
- clear() : SimXMLDocument, Forest, GuiTreeViewCtrl, GuiPopUpMenuCtrlEx, GuiTextListCtrl, MessageVector, PfxVis, SimSet
- clearAim() : AIPlayer
- clearCameraObject() : GameConnection
- clearControlObject() : Player
- clearEntry() : GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
- clearError() : SimXMLDocument
- clearFilterText() : GuiTreeViewCtrl
- clearFirstResponder() : GuiControl
- clearItemColor() : GuiListBoxCtrl
- clearItems() : GuiListBoxCtrl
- clearMenuItems() : GuiMenuBar
- clearMenus() : GuiMenuBar
- clearPaths() : NetConnection
- clearScopeToClient() : NetObject
- clearSelectedText() : GuiTextEditCtrl
- clearSelection() : GuiListBoxCtrl, GuiTextListCtrl, GuiTreeViewCtrl
- clearShaderMacros() : PostEffect
- clearSubmenuItems() : GuiMenuBar
- click() : GuiFadeinBitmapCtrl
- clientToScreen() : GuiCanvas
- clone() : SimObject
- close() : FileStreamObject, FileObject
- closeArchive() : ZipObject
- closeFile() : ZipObject
- collapse() : GuiRolloutCtrl
- computeSizes() : GuiScrollCtrl
- connect() : TCPObject, NetConnection
• connectLocal() : NetConnection
• controlsChild() : GuiControl
• copyFrom() : StreamObject
• count() : ArrayObject
• countKey() : ArrayObject
• countValue() : ArrayObject
• createNullDevice() : GFXInit
• crop() : ArrayObject
• cursorOff() : GuiCanvas
• cursorOn() : GuiCanvas

Copyright © GarageGames, LLC. All Rights Reserved.
- d -

- deactivate() : PhysicalZone, SFXState
- deactivateTurret() : AI_TurretShape
- deepClone() : SimObject
• delete() : SimObject, GameConnection
• deleteAllObjects() : SimSet
• deleteFile() : ZipObject
• deleteItem() : GuiListBoxCtrl
• deleteLine() : MessageVector
• deleteSelection() : GuiTreeViewCtrl
• destroy() : PhysicsShape
• destroyThread() : ShapeBase
• detach() : GuiMessageVectorCtrl, PhysicsForce, ConsoleLogger
• disable() : RenderPassStateToken, PostEffect, SFXState
• disconnect() : TCPObject
• doDismount() : PlayerData
• doMirror() : GuiListBoxCtrl
• doRespawn() : TurretShape
• drawBox() : DebugDrawer
• drawLine() : DebugDrawer
• dump() : MessageVector, SimObject, MessageVector
• dumpClassHierarchy() : SimObject
• dumpEvents() : EventManager
• dumpGroupHierarchy() : SimObject
• dumpMeshVisibility() : ShapeBase
• dumpMethods() : SimObject
• dumpShaderDisassembly() : PostEffect
• dumpShape() : TSShapeConstructor
• dumpSubscribers() : EventManager
• dumpZoneState() : Zone
• duplicate() : ArrayObject

Copyright © GarageGames, LLC. All Rights Reserved.
• All
• Functions
• Variables

• a
• b
• c
• d
• e
• f
• g
• h
• i
• l
• m
• n
• o
• p
• q
• r
• s
• t
• u
• v
• w

- e -

• echo() : ArrayObject
• editItem() : GuiTreeViewCtrl
• elementValue() : SimXMLDocument
- empty() : ArrayObject
- enable() : RenderPassStateToken, PostEffect, SFXState
- erase() : ArrayObject
- Execute() : FileDialog
- expand() : GuiRolloutCtrl
- expandItem() : GuiTreeViewCtrl
- explode() : ProximityMine
- export() : BanList
- exportHeightMap() : TerrainBlock
- exportLayerMaps() : TerrainBlock
- exportToCollada() : InteriorInstance
- extractFile() : ZipObject

Copyright © GarageGames, LLC. All Rights Reserved.
- f -

- findChildItemByName() : GuiTreeViewCtrl
- findHitControl() : GuiControl
- findHitControls() : GuiControl
• findItemByName() : GuiTreeViewCtrl
• findItemById() : GuiTreeViewCtrl
• findItemByIdValue() : GuiTreeViewCtrl
• findItemByText() : GuiListBoxCtrl
• findObjectByIdInternalName() : SimSet
• findText() : GuiPopupMenuCtrl, GuiPopUpMenuCtrlEx
• findTextIndex() : GuiTextListCtrl
• firstAttribute() : SimXMLDocument
• forceClose() : GuiPopUpMenuCtrlEx
• forceOnAction() : GuiPopUpMenuCtrlEx
• forceReflow() : GuiMLTextCtrl
• forceValidateText() : GuiTextEditCtrl
• frameBorder() : GuiFrameSetCtrl
• frameMinExtent() : GuiFrameSetCtrl
• frameMovable() : GuiFrameSetCtrl
• framePadding() : GuiFrameSetCtrl
• freeData() : SFXFMODEventGroup
• freeReference() : Message
• freeze() : GuiStackControl
• freezeSim() : RigidShape
- g -

- get() : HTTPObject
- getAdapterCount() : GFXInit
- getAdapterMode() : GFXInit
- getAdapterModeCount() : GFXInit
- getAdapterName() : GFXInit
- getAdapterShaderModel() : GFXInit
- getAdapterType() : GFXInit
- getAddress() : NetConnection
- getAimLocation() : AIPlayer
- getAimObject() : AIPlayer
- getAlRepairPoint() : ShapeBase
- getAllowManualFire() : TurretShape
- getAllowManualRotation() : TurretShape
- getAngularVelocity() : Camera
- getArea() : MissionArea
- getAspect() : GuiControl
- getAspectRatio() : PostEffect
- getAttenuatedVolume() : SFXSource
- getBinding() : ActionMap
- getBinType() : RenderBinManager
- getBounds() : TSShapeConstructor
- getCameraFov() : ShapeBase
- getCameraObject() : GameConnection
- getCameraSpeed() : GuiObjectView
- getCanSave() : SimObject
- getCard() : GFXCardProfilerAPI
- getCenter() : GuiControl
- getChild() : GuiTreeviewCtrl
- getClassName() : SimObject
- getClassNamespace() : SimObject
- getClientObject() : NetObject
- getColumnCount() : GuiFrameSetCtrl
- getColumnOffset() : GuiFrameSetCtrl
- getCommand() : ActionMap
- getContent() : GuiCanvas
- getControlCameraDefaultFov() : GameConnection
- getControlCameraFov() : GameConnection
- getControllingClient() : ShapeBase
- getControllingObject() : ShapeBase
- getControlObject() : GameConnection, Player
- getCount() : SimSet
- getCurrent() : ArrayObject
- getCurrentLanguage() : LangTable
- getCurrentOption() : GuiGameListOptionsCtrl
- getCurrentSlot() : SFXController
- getCurrentTime() : GuiTheoraCtrl
- getCursorPos() : GuiTextEditCtrl, GuiCanvas
- getDamageFlash() : ShapeBase, GameConnection
- getDamageLevel() : ShapeBase
- getDamageLocation() : Player
- getDamagePercent() : ShapeBase
- getDamageState() : ShapeBase
- getData() : SimXMLDocument
- getDataBlock() : GameBase
- getDatum() : GuiGraphCtrl
- getDeadZone() : ActionMap
- getDebugInfo() : SimObject
- getDeclarationLine() : SimObject
- getDefaultAdapterIndex() : GFXInit
- getDefaultCameraFov() : ShapeBase
- getDeployTransform() : ShapeBaseData
- getDetailLevelCount() : TSShapeConstructor
- getDetailLevelIndex() : TSShapeConstructor
- getDetailLevelName() : TSShapeConstructor
- getDetailLevelSize() : TSShapeConstructor
- getDuration() : SFXSound
- getDynamicField() : SimObject
- getDynamicFieldCount() : SimObject
- getEnergyLevel() : ShapeBase
- getEnergyPercent() : ShapeBase
- getErrorDesc() : SimXMLDocument
- getEulerRotation() : SceneObject
- getExtent() : GuiControl, GuiCanvas
- getEyePoint() : ShapeBase
- getEyeTransform() : ShapeBase
- getEyeVector() : ShapeBase
- getFadeInTime() : SFXSource
- getFadeOutTime() : SFXSource
- getField() : SimObject
- getFieldCount() : SimObject
- getFieldType() : SimObject
• `getFieldValue()` : `SimObject`
• `getFileEntry()` : `ZipObject`
• `getFileEntryCount()` : `ZipObject`
• `getFilename()` : `SimObject`, `CubemapData`
• `getFilterText()` : `GuiTreeViewCtrl`
• `getFirstResponder()` : `GuiControl`
• `getForwardVector()` : `SceneObject`
• `getFramePadding()` : `GuiFrameSetCtrl`
• `getFullCount()` : `SimSet`
• `getGhostID()` : `NetObject`, `NetConnection`
• `getGhostsActive()` : `NetConnection`
• `getGlobalCenter()` : `GuiControl`
• `getGlobalPosition()` : `GuiControl`
• `getGroup()` : `SimObject`
• `getId()` : `SimObject`
• `getImageAltTrigger()` : `ShapeBase`
• `getImageAmmo()` : `ShapeBase`
• `getImageGenericTrigger()` : `ShapeBase`
• `getImageLoaded()` : `ShapeBase`
• `getImageScriptAnimPrefix()` : `ShapeBase`
• `getImageSkinTag()` : `ShapeBase`
• `getImageState()` : `ShapeBase`
• `getImageTarget()` : `ShapeBase`
• `getImageTrigger()` : `ShapeBase`
• `getImposterDetailLevel()` : `TSShapeConstructor`
• `getImposterSettings()` : `TSShapeConstructor`
• `getIndexFromKey()` : `ArrayObject`
• `getIndexFromValue()` : `ArrayObject`
• `getInternalName()` : `SimObject`
• `getInverseTransform()` : `SceneObject`
• `getItemCount()` : `GuiListBoxCtrl`
• `getItemObject()` : `GuiListBoxCtrl`
• `getItemText()` : `GuiListBoxCtrl`, `GuiTreeViewCtrl`
• `getItemValue()` : `GuiTreeViewCtrl`
• `getKey()` : `ArrayObject`
• `getLangName()` : `LangTable`
• `getLastClickItem()` : `GuiListBoxCtrl`
• `getLastStickyNormal()` : `Item`
• `getLastStickyPos()` : `Item`
- `getLineIndexByTag()`: `MessageVector`
- `getLineTag()`: `MessageVector`
- `getLineText()`: `MessageVector`
- `getLineTextByTag()`: `MessageVector`
- `getLookAtPoint()`: `ShapeBase`
- `getManager()`: `RenderPassManager`
- `getManagerCount()`: `RenderPassManager`
- `getMaxDamage()`: `ShapeBase`
- `getMeshCount()`: `TSShapeConstructor`
- `getMeshMaterial()`: `TSShapeConstructor`
- `getMeshName()`: `TSShapeConstructor`
- `getMeshSize()`: `TSShapeConstructor`
- `getMeshType()`: `TSShapeConstructor`
- `getMinExtent()`: `GuiControl`
- `getMode()`: `Camera`, `GuiCanvas`
- `getModeCount()`: `GuiCanvas`
- `getModel()`: `GuiObjectView`
- `getModelFile()`: `ShapeBase`, `TSStatic`, `InteriorInstance`
- `getMountedImage()`: `ShapeBase`
- `getMountedModel()`: `GuiObjectView`
- `getMountedObject()`: `SceneObject`
- `getMountedObjectCount()`: `SceneObject`
- `getMountedObjectNode()`: `SceneObject`
- `getMountNodeObject()`: `SceneObject`
- `getMountSkin()`: `GuiObjectView`
- `getMountSlot()`: `ShapeBase`
- `getMouseControl()`: `GuiCanvas`
- `getMove()`: `AIConnection`
- `getMoveDestination()`: `AIPlayer`
- `getMoveSpeed()`: `AIPlayer`
- `getMuzzlePoint()`: `ShapeBase`
- `getMuzzleVector()`: `ShapeBase`
- `getName()`: `SimObject`
- `getNextSibling()`: ` GuiTreeviewCtrl`
- `getNodeChildCount()`: `TSShapeConstructor`
- `getNodeChildName()`: `TSShapeConstructor`
- `getNodeCount()`: `TSShapeConstructor`
- `getNodeIndex()`: `TSShapeConstructor`
- `getNodeName()`: `TSShapeConstructor`
- getNodeObjectCount() : TSShapeConstructor
- getNodeObjectName() : TSShapeConstructor
- getNodeParentName() : TSShapeConstructor
- getNodeTransform() : TSShapeConstructor
- getNumDeathAnimations() : Player
- getNumDetailLevels() : InteriorInstance
- getNumLang() : LangTable
- getNumLines() : MessageVector
- getNumObjects() : Trigger
- getObject() : SimSet, Trigger
- getObjectBox() : SceneObject
- getObjectCount() : TSShapeConstructor
- getObjectIndex() : TSShapeConstructor, SimSet
- getObjectMount() : SceneObject
- getObjectName() : TSShapeConstructor
- getObjectNode() : TSShapeConstructor
- getOffset() : Camera
- getOrbitDistance() : GuiObjectView
- getPacketLoss() : NetConnection
- getParameter() : SFXSource
- getParameterCount() : SFXSource
- getParameterName() : SFXParameter
- getParent() : GuiControl, GuiTreeViewCtrl
- getPathId() : Path
- getPendingImage() : ShapeBase
- getPing() : NetConnection
- getPitch() : SFXSource
- getPose() : Player
- getPosition() : SceneObject, GuiControl, StreamObject, SFXSound, Camera
- getPrevSibling() : GuiTreeViewCtrl
- getRandom() : SimSet
- getRechargeRate() : ShapeBase
- getRenderer() : GFXCardProfilerAPI
- getRepairRate() : ShapeBase
- getRightVector() : SceneObject
- getRoot() : GuiControl
- getRotation() : Camera
- getRowCount() : GuiFrameSetCtrl, GuiGameListMenuCtrl
- `getRowId()` : `GuiTextListCtrl`
- `getRowLabel()` : `GuiGameListMenuCtrl`
- `getRowNumById()` : `GuiTextListCtrl`
- `getRowOffset()` : `GuiFrameSetCtrl`
- `getRowText()` : `GuiTextListCtrl`
- `getRowTextById()` : `GuiTextListCtrl`
- `getScale()` : `ActionMap`, `SceneObject`
- `getScrollPosition()` : `GuiScrollCtrl`
- `getScrollPositionX()` : `GuiScrollCtrl`
- `getScrollPositionY()` : `GuiScrollCtrl`
- `getSelCount()` : `GuiListBoxCtrl`
- `getSelected()` : `GuiPopUpMenuCtrlEx`
- `getSelectedFile()` : `GuiDirectoryFileListCtrl`
- `getSelectedFiles()` : `GuiDirectoryFileListCtrl`
- `getSelectedId()` : `GuiTextListCtrl`
- `getSelectedItem()` : `GuiTreeViewCtrl`, `GuiListBoxCtrl`
- `getSelectedItems()` : `GuiListBoxCtrl`
- `getSelectedPage()` : `GuiTabBookCtrl`
- `getSelectedRow()` : `GuiGameListMenuCtrl`, `GuiTextListCtrl`
- `getSequenceBlend()` : `TSShapeConstructor`
- `getSequenceCount()` : `TSShapeConstructor`
- `getSequenceCyclic()` : `TSShapeConstructor`
- `getSequenceFrameCount()` : `TSShapeConstructor`
- `getSequenceGroundSpeed()` : `TSShapeConstructor`
- `getSequenceIndex()` : `TSShapeConstructor`
- `getSequenceName()` : `TSShapeConstructor`
- `getSequencePriority()` : `TSShapeConstructor`
- `getSequenceSource()` : `TSShapeConstructor`
- `getServerConnection()` : `GameConnection`
- `getServerObject()` : `NetObject`
- `getShapeName()` : `ShapeBase`
- `getSkin()` : `GuiObjectView`
- `getSkinName()` : `ShapeBase`
- `getSlotTransform()` : `ShapeBase`
- `getSoundDuration()` : `SFXProfile`
- `getSource()` : `SFXEmitter`
- `getState()` : `Player`, `TurretShape`
- `getStatus()` : `SFXSource`, `StreamObject`
- getStreamSize() : StreamObject
- getString() : LangTable
- getStringWidth() : GuiControlProfile
- getSuperClassNamespace() : SimObject
- getTarget() : AITurretShape
- getTargetCount() : TSShapeConstructor, InteriorInstance, TSStatic, ShapeBase
- getTargetName() : TSStatic, ShapeBase, InteriorInstance, TSShapeConstructor
- getText() : GuiTextEditCtrl, GuiButtonBaseCtrl, GuiPopUpMenuCtrlEx, SimXMLDocument, GuiMLTextCtrl
- getTextById() : GuiPopUpMenuCtrlEx, GuiPopUpMenuCtrl
- getTextToRoot() : GuiTreeViewCtrl
- getTime() : GuiClockHud
- getTransform() : SceneObject
- getTrigger() : AlConnection, TSShapeConstructor
- getTriggerCount() : TSShapeConstructor
- getTurretEulerRotation() : TurretShape
- getType() : Message, SceneObject
- getUpVector() : SceneObject
- getValue() : GuiSliderCtrl, ArrayObject
- getVelocity() : Camera, ShapeBase
- getVendor() : GFXCardProfilerAPI
- getVersion() : GFXCardProfilerAPI
- getVideoMemoryMB() : GFXCardProfilerAPI
- getVideoMode() : GuiCanvas
- getVolume() : SFXSource
- getWeaponLeadVelocity() : AITurretShape
- getWheelCount() : WheeledVehicle
- getWhiteOut() : ShapeBase, GameConnection
- getWindowPosition() : GuiCanvas
- getWorldBox() : SceneObject
- getWorldBoxCenter() : SceneObject
- getWorldToScreenScale() : GuiTSCtrl
- getZoneId() : Zone
- h -

- handleRenameObject() : GuiTreeCtrl
- hasImageState() : ShapeBase
- hasTarget() : ALTurretShape
- hide() : PfxVis
- hideCursor() : GuiCanvas
- hideSelection() : GuiTreeCtrl

Copyright © GarageGames, LLC. All Rights Reserved.
- i -
  - import() : TerrainBlock
  - init() : Debris
  - initialControlSet() : GameConnection
- insert() : ArrayObject
- insertItem() : GuiListBoxCtrl, GuiTreeViewCtrl
- insertLine() : MessageVector
- instantCollapse() : GuiRolloutCtrl
- instantExpand() : GuiRolloutCtrl
- isActive() : SFXState
- isAIControlled() : GameConnection
- isAllTextSelected() : GuiTextEditCtrl
- isAtRest() : Item
- isAttached() : PhysicsForce
- isAwake() : GuiControl
- isBanned() : BanList
- isChildOfGroup() : SimObject
- isClientObject() : NetObject
- isCloaked() : ShapeBase
- isControlObjectRotDampedCamera() : GameConnection
- isCursorOn() : GuiCanvas
- isCursorShown() : GuiCanvas
- isDataLoaded() : SFXFMODEventGroup
- isDemoPlaying() : GameConnection
- isDemoRecording() : GameConnection
- isDestroyed() : ShapeBase, PhysicsShape
- isDirty() : Forest
- isDisabled() : ShapeBase, SFXState
- isEditOrbitMode() : Camera
- isEditorOnly() : SimObject
- isEnabled() : ShapeBase, PostEffect
- isEOF() : FileObject, StreamObject
- isEOS() : StreamObject
- isExpanded() : SimObject, GuiRolloutCtrl
- isExteriorPortal() : Portal
- isField() : SimObject
- isFirstPerson() : GameConnection
- isFirstResponder() : GuiControl
- isFrozen() : GuiStackControl
- isFullscreen() : GuiCanvas
- isGlobalBounds() : SceneObject
- isHidden() : ShapeBase
- isImageFiring() : ShapeBase
- isImageMounted() : ShapeBase
- isInNamespaceHierarchy() : SimObject
- isInteriorPortal() : Portal
- isInverted() : ActionMap
- isItemSelected() : GuiTreeViewCtrl
- isMaximized() : GuiCanvas
- isMember() : SimSet
- isMemberOfClass() : SimObject
- isMethod() : SimObject
- isMinimized() : GuiCanvas
- isMounted() : SceneObject
- isMouseLocked() : GuiControl
- isNameChangeAllowed() : SimObject
- isObjectMirrored() : GuiListBoxCtrl
- isParentItem() : GuiTreeViewCtrl
- isPaused() : SFXSource
- isPlaybackDone() : GuiTheoraCtrl
- isPlaying() : SFXSource
- isReady() : SFXSound
- isRegisteredEvent() : EventManager
- isRotating() : Item
- isRotationDamped() : Camera
- isRowActive() : GuiTextListCtrl
- isRowEnabled() : GuiGameListMenuCtrl
- isSelected() : SimObject
- isServerObject() : NetObject
- isStateOn() : GuiCheckBoxCtrl
- isStatic() : Item
- isStopped() : SFXSource
- isThumbBeingDragged() : GuiSliderCtrl
- isValidDragTarget() : GuiTreeViewCtrl
- isVisible() : GuiControl
- All
- Functions
- Variables

- a
- b
- c
- d
- e
- f
- g
- h
- i
- l
- m
- n
- o
- p
- q
- r
- s
- t
- u
- v
- w

- I -

- lastAttribute() : SimXMLDocument
- listClassIDs() : GameConnection
- listen() : TCPObjec
• listMeshes() : PxMultiActor
• listObjects() : SimSet
• loadData() : SFXFMODEventGroup
• loadFile() : SimXMLDocument
• lockSelection() : GuiTreeViewCtrl
• lookAt() : Camera
- m -

- makeFirstResponder() : GuiControl
- markItem() : GuiTreeViewCtrl
- matchScale() : GuiGraphCtrl
- maximizeWindow() : GuiCanvas
- minimizeWindow() : GuiCanvas
- modifyStorm() : Precipitation
- mountImage() : ShapeBase
- mountObject() : SceneObject
- moveFirst() : ArrayObject
- moveItemDown() : GuiTreeViewCtrl
- moveItemUp() : GuiTreeViewCtrl
- moveLast() : ArrayObject
- moveNext() : ArrayObject
- movePrev() : ArrayObject
- msg() : SimpleMessageEvent

Copyright © GarageGames, LLC. All Rights Reserved.
- n -

- nextAttribute() : SimXMLDocument
- nextSiblingElement() : SimXMLDocument
- notifyShapeChanged() : TSShapeConstructor
Main   Class List   Namespace List   Online

- All
- Functions
- Variables

- a
- b
- c
- d
- e
- f
- g
- h
- i
- l
- m
- n
- o
- p
- q
- r
- s
- t
- u
- v
- w

- o -

- onAction() : GuiControl
- onActivate() : SFXState
- onActive() : GuiControl
- `onAdd()`: GameBaseData, Trigger, ScriptObject, GuiControl, ScriptGroup, PostEffect, ScriptMsgListener, Message, SpawnSphere
- `onAddGroupSelected()`: GuiTreeViewCtrl
- `onAddMultipleSelectionBegin()`: GuiTreeViewCtrl
- `onAddMultipleSelectionEnd()`: GuiTreeViewCtrl
- `onAddSelection()`: GuiTreeViewCtrl
- `onAddToQueue()`: ScriptMsgListener
- `onAltClick()`: GuiBitmapButtonCtrl
- `onBeginReparenting()`: GuiTreeViewCtrl
- `onChange()`: GuiGameListMenuCtrl
- `onChildAdded()`: GuiScriptNotifyCtrl
- `onChildRemoved()`: GuiScriptNotifyCtrl
- `onChildResized()`: GuiScriptNotifyCtrl
- `onClearSelection()`: GuiListBoxCtrl, GuiTreeViewCtrl
- `onClick()`: GuiButtonBaseCtrl
- `onClose()`: GuiWindowCtrl
- `onCollapse()`: GuiWindowCtrl
- `onCollapseds()`: GuiRolloutCtrl
- `onCollision()`: ShapeBaseData, ProjectileData
- `onComplete()`: GuiAutoScrollCtrl
- `onConnected()`: TCPObject
- `onConnectFailed()`: TCPObject
- `onConnectionAccepted()`: GameConnection
- `onConnectionDropped()`: GameConnection
- `onConnectionError()`: GameConnection
- `onConnectionRequest()`: TCPObject
- `onConnectionTimedOut()`: GameConnection
- `onConnectRequestRejected()`: GameConnection
- `onConnectRequestTimedOut()`: GameConnection
- `onControlDragEnter()`: GuiControl
- `onControlDragExit()`: GuiControl
- `onControlDragged()`: GuiControl
- `onControlDropped()`: GuiControl
- `onControlObjectChange()`: GameConnection
- `onCtrlClick()`: GuiBitmapButtonCtrl
- `onDamage()`: ShapeBaseData
- `onDataBlocksDone()`: GameConnection
- `onDeactivate()`: SFXState
- `onDefaultClick()` : GuiBitmapButtonCtrl
- `onDefineIcons()` : GuiTreeViewCtrl
- `onDeleteKey()` : GuiListBoxCtrl, GuiTextListCtrl
- `onDeleteObject()` : GuiTreeViewCtrl
- `onDeleteSelection()` : GuiTreeViewCtrl
- `onDestroyed()` : ShapeBaseData
- `onDialogPop()` : GuiControl
- `onDialogPush()` : GuiControl
- `onDisabled()` : ShapeBaseData, PostEffect
- `onDisconnect()` : TCPObject
- `onDNSFailed()` : TCPObject
- `onDNSResolved()` : TCPObject
- `onDone()` : GuiFadeinBitmapCtrl
- `onDoubleClick()` : GuiButtonBaseCtrl, GuiListBoxCtrl
- `onDragDropped()` : GuiTreeViewCtrl
- `onDrop()` : GameConnection
- `onEnabled()` : ShapeBaseData, PostEffect
- `onEndReparenting()` : GuiTreeViewCtrl
- `onEndSequence()` : ShapeBaseData
- `onEnterLiquid()` : VehicleData, Item, PlayerData, RigidShape
- `onEnterMissionArea()` : PlayerData
- `onEnterTrigger()` : TriggerData
- `onExpanded()` : GuiRolloutCtrl
- `onExplode()` : ProjectileData, ProximityMineData
- `onFlash()` : GameConnection
- `onForceUncloak()` : ShapeBaseData
- `onGainFirstResponder()` : GuiControl, GuiScriptNotifyCtrl
- `onHeaderRightClick()` : GuiRolloutCtrl
- `onImpact()` : ShapeBaseData
- `onInputEvent()` : GuiInputCtrl
- `onInspect()` : GuiTreeViewCtrl
- `onKeyDown()` : GuiTreeViewCtrl
- `onLeaveLiquid()` : VehicleData, Item, PlayerData, RigidShape
- `onLeaveMissionArea()` : PlayerData
- `onLeaveTrigger()` : TriggerData
- `onLine()` : TCPObject
- `onLoad()` : Prefab, TSShapeConstructor
- `onLoseFirstResponder()` : GuiControl, GuiScriptNotifyCtrl
- `onMaximize()` : GuiWindowCtrl
- onMenuItemSelect() : GuiMenuBar
- onMenuSelect() : GuiMenuBar
- onMessageObjectReceived() : ScriptMsgListener
- onMessageReceived() : ScriptMsgListener
- onMessageSelected() : GuiConsole
- onMinimize() : GuiWindowCtrl
- onMount() : ShapeBaseImageData, GameBaseData
- onMountObject() : TurretShapeData
- onMouseDown() : GuiButtonBaseCtrl, GuiMouseEventCtrl
- onMouseDragged() : GuiMouseEventCtrl, GuiButtonBaseCtrl, GuiListBoxCtrl, GuiTreeViewCtrl, GuiSliderCtrl
- onMouseEnter() : GuiButtonBaseCtrl, GuiObjectView, GuiMouseMoveEventCtrl
- onMouseInMenu() : GuiMenuBar
- onMouseLeave() : GuiButtonBaseCtrl, GuiObjectView, GuiMouseMoveEventCtrl
- onMouseMove() : GuiMouseMoveEventCtrl
- onMouseUp() : GuiButtonBaseCtrl, GuiListBoxCtrl, GuiTreeViewCtrl, GuiMouseMoveEventCtrl
- onNewDataBlock() : GameBaseData
- onNode() : PathCamera
- onObjectAdded() : SimSet
- onObjectDeleteCompleted() : GuiTreeViewCtrl
- onObjectRemoved() : SimSet
- onParameterValueChange() : SFXSource
- onParentResized() : GuiScriptNotifyCtrl
- onPoseChange() : PlayerData
- onRemove() : GuiControl, Message, GameBaseData, ScriptObject, ScriptGroup, Trigger, ScriptMsgListener
- onRemoveFromQueue() : ScriptMsgListener
- onRemoveSelection() : GuiTreeViewCtrl
- onReparent() : GuiTreeViewCtrl
- onReset() : GuiAutoScrollCtrl
- onResize() : GuiScriptNotifyCtrl, GuiMLTextCtrl
- onRestore() : GuiWindowCtrl
- onReturn() : GuiTextEditCtrl
- onRightClick() : GuiButtonBaseCtrl
- onRightMouseDown() : GuiTreeViewCtrl, GuiMouseEventCtrl
- onRightMouseDragged() : GuiMouseMoveEventCtrl
onRightMouseUp() : GuiMouseEventHandler, GuiTreeViewCtrl
onScroll() : GuiScrollCtrl
onSelect() : GuiTextListCtrl, GuiTreeViewCtrl, GuiListBoxCtrl
onShiftClick() : GuiBitmapButtonCtrl
onSleep() : GuiControl
onStart() : GuiAutoScrollCtrl
onStartSprintMotion() : PlayerData
onStartSwim() : PlayerData
onStatusChange() : SFXSource
onStickyCollision() : Item, TurretShapeData
onStopSprintMotion() : PlayerData
onStopSwim() : PlayerData
onSubmenuSelect() : GuiMenuBar
onTabComplete() : GuiTextEditCtrl
onTabRightClick() : GuiTabBookCtrl
onTabSelected() : GuiTabBookCtrl
onTick() : GuiAutoScrollCtrl
onTickTrigger() : TriggerData
onTrigger() : ShapeBaseData
onTriggered() : ProximityMineData
onUnload() : TSShapeConstructor
onUnmount() : ShapeBaseImageData, GameBaseData
onUnmountObject() : TurretShapeData
onUnselect() : GuiTreeViewCtrl, GuiListBoxCtrl
onUpdate() : SFXParameter
onURL() : GuiMLTextCtrl
onValidate() : GuiTextEditCtrl
onVisible() : GuiControl
onWake() : GuiControl
onWindowClosed() : PfxVis
open() : FileStreamObject, GuiTreeViewCtrl, PfxVis
openArchive() : ZipObject
openFileForRead() : ZipObject
openFileForWrite() : ZipObject
openForAppend() : FileObject
openForRead() : FileObject
openForWrite() : FileObject
- p -

- parse(): SimXMLDocument
- pause(): SFXSource, GuiTheoraCtrl
- pauseThread(): ShapeBase
- peekLine() : FileObject
- performClick() : GuiButtonBaseCtrl
- play() : GuiTheoraCtrl, SFXEmitter
- play2D() : GameConnection
- play3D() : GameConnection
- playAnimation() : LightBase
- playAudio() : ShapeBase
- playDemo() : GameConnection
- pointInControl() : GuiControl
- pop() : ActionMap
- pop_back() : ArrayObject
- pop_front() : ArrayObject
- popBackLine() : MessageVector
- popDialog() : GuiCanvas
- popElement() : SimXMLDocument
- popFront() : PathCamera
- popFrontLine() : MessageVector
- popLayer() : GuiCanvas
- post() : HTTPObject
- postApply() : RenderMeshExample, DecalRoad, MeshRoad, DecalData, GroundPlane, MissionArea
- postEvent() : EventManager
- preProcess() : PostEffect
- presimulate() : Projectile
- prevAttribute() : SimXMLDocument
- project() : GuiTSCtrl
- push() : ActionMap
- push_back() : ArrayObject
- push_front() : ArrayObject
- pushBack() : PathCamera
- pushBackLine() : MessageVector
- pushChildElement() : SimXMLDocument
- pushDialog() : GuiCanvas
- pushFirstChildElement() : SimXMLDocument
- pushFront() : PathCamera
- pushFrontLine() : MessageVector
- pushNewElement() : SimXMLDocument
- pushToBack() : SimSet
- All 
  - Functions 
  - Variables 

- a 
- b 
- c 
- d 
- e 
- f 
- g 
- h 
- i 
- l 
- m 
- n 
- o 
- p 
- q 
- r 
- s 
- t 
- u 
- v 
- w 

- q -
  - queryProfile() : GFXCardProfilerAPI
- r -

- readComment() : SimXMLDocument
- readLine() : FileObject, StreamObject
- readLongString() : StreamObject
- `readString()`: `StreamObject`
- `readSTString()`: `StreamObject`
- `recenterTurret()`: `AITurretShape`
- `refresh()`: `GuiDynamicCtrlArrayControl`
- `regenCells()`: `Forest`
- `regenerate()`: `DecalRoad, MeshRoad, River`
- `registerEvent()`: `EventManager`
- `reload()`: `GuiDirectoryFileListCtrl, ShaderData, PostEffect, ParticleData, ParticleEmitterData`
- `reloadOnLocalClient()`: `SimDataBlock`
- `remove()`: `SimSet, EventManager`
- `removeAll()`: `EventManager`
- `removeAutoPlot()`: `GuiGraphCtrl`
- `removeBan()`: `BanList`
- `removeColumn()`: `GuiFrameSetCtrl`
- `removeDetailLevel()`: `TSShapeConstructor`
- `removeFilteredItem()`: `GuiListBoxCtrl`
- `removeFromIgnoreList()`: `AITurretShape`
- `removeImposter()`: `TSShapeConstructor`
- `removeItem()`: `GuiTreeViewCtrl`
- `removeManager()`: `RenderPassManager`
- `removeMenu()`: `GuiMenuBar`
- `removeMenuItem()`: `GuiMenuBar`
- `removeMesh()`: `TSShapeConstructor`
- `removeNode()`: `TSShapeConstructor`
- `removeObject()`: `TSShapeConstructor`
- `removeParameter()`: `SFXSource`
- `removeRow()`: `GuiFrameSetCtrl, GuiTextListCtrl`
- `removeRowById()`: `GuiTextListCtrl`
- `removeSelection()`: `GuiTreeViewCtrl`
- `removeSequence()`: `TSShapeConstructor`
- `removeShaderMacro()`: `PostEffect`
- `removeText()`: `SimXMLDocument`
- `removeTrigger()`: `TSShapeConstructor`
- `renameDetailLevel()`: `TSShapeConstructor`
- `renameNode()`: `TSShapeConstructor`
- `renameObject()`: `TSShapeConstructor`
- `renameSequence()`: `TSShapeConstructor`
- `renderFront()`: `GuiCanvas`
- reorderChild() : SimSet
- repaint() : GuiCanvas
- replaceText() : GuiPopUpMenuCtrl
- reset() : SFXParameter, PathCamera, RigidShape, GuiCanvas, SimXMLDocument, GuiAutoScrollCtrl
- resetGhosting() : GameConnection
- resetState() : GuiButtonBaseCtrl
- resetTarget() : AiTurretShape
- resize() : GuiControl
- resolveGhostID() : NetConnection
- resolveObjectFromGhostIndex() : NetConnection
- restore() : PhysicsShape
- restoreWindow() : GuiCanvas
- rowCount() : GuiTextListCtrl

Copyright © GarageGames, LLC. All Rights Reserved.
- s -

- save() : SimObject, ActionMap, TerrainBlock
- saveFile() : SimXMLDocument
- saveShape() : TSShapeConstructor
- `schedule()`: `SimObject`
- `scopeToClient()`: `NetObject`
- `screenToClient()`: `GuiCanvas`
- `scrollToBottom()`: `GuiMLTextCtrl`, `GuiScrollCtrl`
- `scrollToObject()`: `GuiScrollCtrl`
- `scrollToTag()`: `GuiMLTextCtrl`
- `scrollToTop()`: `GuiMLTextCtrl`, `GuiScrollCtrl`
- `scrollVisible()`: `GuiTextListCtrl`, `GuiTreeViewCtrl`
- `scrollVisibleByObjectId()`: `GuiTreeViewCtrl`
- `select()`: `GuiTabPageCtrl`
- `selectAllText()`: `GuiTextEditCtrl`
- `selectItem()`: `GuiTreeViewCtrl`
- `selectOption()`: `GuiGameListOptionsCtrl`
- `selectPage()`: `GuiTabBookCtrl`
- `selectWindow()`: `GuiWindowCtrl`
- `send()`: `RadialImpulseEvent`, `TCPObject`
- `setActionThread()`: `Player`
- `setActive()`: `GuiControl`, `ParticleEmitterNode`
- `setAimLocation()`: `AIPlayer`
- `setAimObject()`: `AIPlayer`
- `setAlarmMode()`: `InteriorInstance`
- `setAllGunsFiring()`: `AITurretShape`
- `setAllHidden()`: `PxMultiActor`
- `setAllMeshesHidden()`: `ShapeBase`
- `setAllowManualFire()`: `TurretShape`
- `setAllowManualRotation()`: `TurretShape`
- `setAlpha()`: `GuiMLTextCtrl`
- `setAngularDrag()`: `Camera`
- `setAngularForce()`: `Camera`
- `setAngularVelocity()`: `Camera`
- `setArea()`: `MissionArea`
- `setArmThread()`: `Player`
- `setAttribute()`: `SimXMLDocument`
- `setBatchSize()`: `River`
- `setBitmap()`: `GuiBitmapCtrl`, `GuiBitmapButtonCtrl`, `GuiIconButtonCtrl`, `GuiChunkedBitmapCtrl`, `GuiProgressBitmapCtrl`
- `setBlackOut()`: `GameConnection`
- `setBounds()`: `TSShapeConstructor`
- `setBrakeMultiplier()`: `Camera`
- `setBroken()`: `PxMultiActor`
- `setCameraFov()`: `ShapeBase`
- `setCameraObject()`: `GameConnection`
- `setCameraSpeed()`: `GuiObjectView`
- `setCanSave()`: `SimObject`
- `setCapability()`: `GFXCardProfilerAPI`
- `setCenter()`: `GuiControl`
- `setCheckmarkBitmapIndex()`: `GuiMenuBar`
- `setClassNamespace()`: `SimObject`
- `setCloaked()`: `ShapeBase`
- `setCollapsed()`: `GuiPaneControl`
- `setCollapseGroup()`: `GuiWindowCtrl`
- `setCollisionTimeout()`: `Item`
- `setColor()`: `GuiSwatchButtonCtrl`
- `setColumnOffset()`: `GuiFrameSetCtrl`
- `setCone()`: `SFXSource`
- `setConnectArgs()`: `GameConnection`
- `setContent()`: `GuiCanvas`
- `setControl()`: `GameBase`
- `setControlCameraFov()`: `GameConnection`
- `setControlObject()`: `GameConnection, Player`
- `setCurrent()`: `ArrayObject`
- `setCurrentLanguage()`: `LangTable`
- `setCurrentSlot()`: `SFXController`
- `setCurSel()`: `GuiListBoxCtrl`
- `setCurSelRange()`: `GuiListBoxCtrl`
- `setCursor()`: `GuiCanvas`
- `setCursorPos()`: `GuiTextEditCtrl, GuiCanvas`
- `setCursorPosition()`: `GuiMLTextCtrl`
- `setDamageFlash()`: `ShapeBase`
- `setDamageLevel()`: `ShapeBase`
- `setDamageState()`: `ShapeBase`
- `setDamageVector()`: `ShapeBase`
- `setDataBlock()`: `GameBase`
- `setDayLength()`: `TimeOfDay`
- `setDebug()`: `GuiTreeViewCtrl`
- `setDefaultLanguage()`: `LangTable`
- `setDetailLevel()`: `InteriorInstance`
- `setDetailLevelSize()`: `TSShapeConstructor`
- `setDrag()`: `Camera`
- `setEditOrbitMode()`: `Camera`
- `setEditOrbitPoint()`: `Camera`
- `setEditorOnly()`: `SimObject`
- `setEmitterDataBlock()`: `ParticleEmitterNode`
- `setEnergyLevel()`: `ShapeBase`
- `setEnumContent()`: `GuiPopUpMenuCtrl`
- `setExtent()`: `GuiControl`
- `setFadeTimes()`: `SFXSource`
- `setFieldType()`: `SimObject`
- `setFieldValue()`: `SimObject`
- `setFile()`: `GuiTheoraCtrl`
- `setFilename()`: `SimObject`
- `setFilter()`: `GuiDirectoryFileListCtrl`
- `setFilterText()`: `GuiTreeViewCtrl`
- `setFirstPerson()`: `GameConnection`
- `setFirstResponder()`: `GuiControl`
- `setFirstSelected()`: `GuiPopUpMenuCtrl`
- `setFlyForce()`: `Camera`
- `setFlyMode()`: `Camera`
- `setFocus()`: `GuiCanvas`
- `setFreeLook()`: `AIConnection`
- `setGraphType()`: `GuiGraphCtrl`
- `setGunSlotFiring()`: `AITurretShape`
- `setHidden()`: `SimObject`, `ShapeBase`
- `setImageAltTrigger()`: `ShapeBase`
- `setImageAmmo()`: `ShapeBase`
- `setImageGenericTrigger()`: `ShapeBase`
- `setImageLoaded()`: `ShapeBase`
- `setImageScriptAnimPrefix()`: `ShapeBase`
- `setImageTarget()`: `ShapeBase`
- `setImageTrigger()`: `ShapeBase`
- `setInternalName()`: `SimObject`
- `setInvincibleMode()`: `ShapeBase`
- `setIsExpanded()`: `SimObject`
- `setIsSelected()`: `SimObject`
- `setItemColor()`: `GuiListBoxCtrl`
- `setItemImages()`: `GuiTreeViewCtrl`
- `setItemText()` : `GuiListBoxCtrl`
- `setItemTooltip()` : `GuiListBoxCtrl`, `GuiTreeCtrl`
- `setJoinPassword()` : `GameConnection`
- `setKey()` : `ArrayOfObject`
- `setLagIcon()` : `GameConnection`
- `setLastTTL()` : `DebugDrawer`
- `setLastZTest()` : `DebugDrawer`
- `setLightAmbient()` : `GuiObjectView`
- `setLightColor()` : `GuiObjectView`
- `setLightDirection()` : `GuiObjectView`
- `setLightEnabled()` : `LightBase`
- `setLocked()` : `SimObject`
- `setMass()` : `Camera`
- `setMaxDivisionSize()` : `River`
- `setMenuBitmapIndex()` : `GuiMenuBar`
- `setMenuItemBitmap()` : `GuiMenuBar`
- `setMenuItemChecked()` : `GuiMenuBar`
- `setMenuItemEnable()` : `GuiMenuBar`
- `setMenuItemSubmenuState()` : `GuiMenuBar`
- `setMenuItemText()` : `GuiMenuBar`
- `setMenuItemVisible()` : `GuiMenuBar`
- `setMenuMargins()` : `GuiMenuBar`
- `setMenuText()` : `GuiMenuBar`
- `setMenuVisible()` : `GuiMenuBar`
- `setMeshHidden()` : `ShapeBase`, `PxMultiActor`
- `setMeshMaterial()` : `TSShapeConstructor`
- `setMeshSize()` : `TSShapeConstructor`
- `setMeshType()` : `TSShapeConstructor`
- `setMessage()` : `SimpleNetObject`
- `setMetersPerSegment()` : `River`
- `setMissionCRC()` : `GameConnection`
- `setModel()` : `GuiObjectView`
- `setMount()` : `GuiObjectView`
- `setMountedModel()` : `GuiObjectView`
- `setMountSkin()` : `GuiObjectView`
- `setMove()` : `AIConnection`
- `setMoveDestination()` : `AIPlayer`
- `setMoveSpeed()` : `AIPlayer`
- `setMultipleSelection()` : `GuiListBoxCtrl`
- setName() : SimObject
- setNameChangeAllowed() : SimObject
- setNewtonFlyMode() : Camera
- setNodeDepth() : MeshRoad, River
- setNodeParent() : TSShapeConstructor
- setNodeTransform() : TSShapeConstructor
- setNoneSelected() : GuiPopUpMenuCtrlEx
- setObjectAttributes() : SimXMLDocument
- setObjectNode() : TSShapeConstructor
- setOffset() : Camera
- setOptions() : GuiGameListOptionsCtrl
- setOrbitDistance() : GuiObjectView
- setOrbitMode() : Camera
- setOrbitObject() : Camera
- setOrbitPoint() : Camera
- setPath() : GuiDirectoryFileListCtrl
- setPercentage() : Precipitation
- setPitch() : SFXSource
- setPlay() : TimeOfDay
- setPosition() : GuiControl, PathCamera, SFXSound, StreamObject
- setPositionGlobal() : GuiControl
- setProcessTicks() : GuiTickCtrl
- setProfile() : GuiControl
- setRechargeRate() : ShapeBase
- setRepairRate() : ShapeBase
- setReverseTime() : GuiClockHud
- setRotation() : Camera
- setRowActive() : GuiTextListCtrl
- setRowById() : GuiTextListCtrl
- setRowEnabled() : GuiGameListMenuCtrl
- setRowLabel() : GuiGameListMenuCtrl
- setRowOffset() : GuiFrameSetCtrl
- setScale() : SceneObject
- setScopeAlways() : NetObject
- setScrollPosition() : GuiScrollCtrl
- setSelected() : GuiPopUpMenuCtrlEx, GuiListBoxCtrl, GuiGameListMenuCtrl, GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
• setSelectedById() : GuiTextListCtrl
• setSelectedRow() : GuiTextListCtrl
• setSeq() : GuiObjectView
• setSequenceBlend() : TSShapeConstructor
• setSequenceCyclic() : TSShapeConstructor
• setSequenceGroundSpeed() : TSShapeConstructor
• setSequencePriority() : TSShapeConstructor
• setShaderConst() : PostEffect
• setShaderConsts() : PostEffect
• setShaderMacro() : PostEffect
• setShapeName() : ShapeBase
• setSimulatedNetParams() : NetConnection
• setSkin() : GuiObjectView
• setSkinName() : ShapeBase
• setSpeedMultiplier() : Camera
• setState() : PathCamera
• setStateOn() : GuiCheckBoxCtrl, GuiButtonBaseCtrl
• setSubmenuItemChecked() : GuiMenuBar
• setSuperClassNamespace() : SimObject
• setTarget() : PathCamera
• setText() : GuiTextCtrl, GuiMLTextCtrl, GuiPopUpMenuCtrlEx, GuiButtonBaseCtrl, GuiTextEditCtrl
• setTextID() : GuiButtonBaseCtrl, GuiTextCtrl
• setTexture() : PostEffect
• setThreadDir() : ShapeBase
• setThreadPosition() : ShapeBase
• setThreadTimeScale() : ShapeBase
• setTime() : GuiClockHud
• setTimeOfDay() : TimeOfDay
• setTrackObject() : Camera
• setTransform() : SceneObject, SFXSource
• setTrigger() : AIClone
• setTurbulence() : Precipitation
• setTurretEulerRotation() : TurretShape
• setTurretState() : AITurretShape
• setValidEditOrbitPoint() : Camera
• setValue() : GuiBitmapCtrl, ArrayObject, GuiControl, GuiSliderCtrl
• setVelocity() : Camera, ShapeBase
setVideoMode(): GuiCanvas
setVisible(): GuiControl
setVolume(): SFXSource
setWeaponLeadVelocity(): AITurretShape
setWheelPowered(): WheeledVehicle
setWheelSpring(): WheeledVehicle
setWheelSteering(): WheeledVehicle
setWheelTire(): WheeledVehicle
setWhiteOut(): ShapeBase
setWindowPosition(): GuiCanvas
setWindowTitle(): GuiCanvas
show(): PfxVis
showCursor(): GuiCanvas
showItemRenameCtrl(): GuiTreeViewCtrl
sizeToContents(): GuiRolloutCtrl
sort(): GuiTextListCtrl, GuiTreeViewCtrl, SimSet, GuiPopUpMenuCtrlEx, ArrayObject
sorta(): ArrayObject
sortd(): ArrayObject
sortf(): ArrayObject
sortfd(): ArrayObject
sortfk(): ArrayObject
sortfkd(): ArrayObject
sortByID(): GuiPopUpMenuCtrlEx
sortk(): ArrayObject
sortka(): ArrayObject
sortkd(): ArrayObject
sortn(): ArrayObject
sortna(): ArrayObject
sortnd(): ArrayObject
sortnk(): ArrayObject
sortnka(): ArrayObject
sortnkd(): ArrayObject
sortNumerical(): GuiTextListCtrl
spawnObject(): SpawnSphere
startDragging(): GuiDragAndDropControl
startFade(): ShapeBase
startRecording(): GameConnection
startScanForTargets(): AITurretShape
- `startTrackingTarget()` : `AITurretShape`
- `stop()` : `SFXEmitter`, `AIPlayer`, `GuiTheoraCtrl`, `SFXSource`
- `stopAudio()` : `ShapeBase`
- `stopRecording()` : `GameConnection`
- `stopScanForTargets()` : `AITurretShape`
- `stopThread()` : `ShapeBase`
- `stopTrackingTarget()` : `AITurretShape`
- `strikeObject()` : `Lightning`
- `strikeRandomPoint()` : `Lightning`
- `subscribe()` : `EventManager`
- t -

- toggle() : RenderPassStateToken, PostEffect
- toggleCollapse() : GuiRolloutCtrl
- toggleCollapseGroup() : GuiWindowCtrl
- toggleDrawing() : DebugDrawer
- toggleExpanded() : GuiRolloutCtrl
- toggleFreeze() : DebugDrawer
- toggleFullscreen() : GuiCanvas
- toggleHideSelection() : GuiTreeviewCtrl
- toggleLockSelection() : GuiTreeviewCtrl
- transmitDataBlocks() : GameConnection
- transmitPaths() : NetConnection
- u -

- `unbind()` : `ActionMap`
- `unbindObj()` : `ActionMap`
- `uniqueKey()` : `ArrayObject`
• uniqueValue() : ArrayObject
• unmount() : SceneObject
• unmountImage() : ShapeBase
• unmountObject() : SceneObject
• unproject() : GuiTSCtrl
• unregisterEvent() : EventManager
• updateFaces() : CubemapData
• updateSizes() : GuiFrameSetCtrl
• updateStack() : GuiStackControl
• useCreateHeight() : FlyingVehicle
- v -

- validateCameraFov() : ShapeBase
- w -

- warningFlashes() : Lightning
- writeChangeSet() : TSShapeConstructor
- writeLine() : StreamObject, FileObject
• writeLongString() : StreamObject
• writeObject() : FileObject
• writeString() : StreamObject
- b -
- backgroundColor : GuiTheoraCtrl
- backSidePassable : Portal
- backwardJetEmitter : FlyingVehicleData
- barBehindText : GuiPaneControl
- baseColor : CloudLayer, WaterObject
- baseRadius : DebrisData
- baseShape : TSShapeConstructor
- baseTex : Material
- baseTexSize : TerrainBlock
- bending : PxCloth
- bendingStiffness : PxCloth
- bevelColorHL : GuiControlProfile
- bevelColorLL : GuiControlProfile
- billboardUVs : GroundCover
- binType : RenderBinManager
- bitmap : GuiBitmapCtrl, GuiBitmapButtonCtrl, GuiControlProfile
  , GuiTextEditSliderBitmapCtrl, GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx, GuiChunkedBitmapCtrl, GuiProgressBitmapCtrl
- bitmapBounds : GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
- bitmapMode : GuiBitmapButtonCtrl
- bitmapName : GuiCursor
- blendDefined : GFXStateBlockData
- blendDest : GFXStateBlockData
- blendEnable : GFXStateBlockData
- blendOp : GFXStateBlockData
- blendSrc : GFXStateBlockData
- blendStyle : ParticleEmitterData
- bodyFriction : RigidShapeData, VehicleData
- bodyRestitution : RigidShapeData, VehicleData
- boltStartRadius : Lightning
- border : GuiControlProfile
- borderColor : GuiControlProfile, GuiFrameSetCtrl
- borderColorHL : GuiControlProfile
- borderColorNA : GuiControlProfile
- borderEnable : GuiFrameSetCtrl
- borderMargin : GuiSeparatorCtrl
- borderMovable : GuiFrameSetCtrl
- borderThickness : GuiControlProfile
- `borderWidth` : `GuiFrameSetCtrl`
- `bottomMaterial` : `MeshRoad`
- `bounceElasticity` : `ProjectileData`
- `bounceFriction` : `ProjectileData`
- `bounceVariance` : `DebrisData`
- `boundingBox` : `PlayerData`, `GameBase`
- `boxHeadBackPercentage` : `PlayerData`
- `boxHeadFrontPercentage` : `PlayerData`
- `boxHeadLeftPercentage` : `PlayerData`
- `boxHeadPercentage` : `PlayerData`
- `boxHeadRightPercentage` : `PlayerData`
- `boxHeight` : `Precipitation`
- `boxTorsoPercentage` : `PlayerData`
- `boxWidth` : `Precipitation`
- `brakeMultiplier` : `Camera`
- `brakeTorque` : `WheeledVehicleData`
- `brakingActivationSpeed` : `HoverVehicleData`
- `brakingForce` : `HoverVehicleData`
- `branchAmp` : `ForestItemData`
- `breakAngle` : `MeshRoad`, `DecalRoad`
- `breakForce` : `PxMultiActorData`
- `brightness` : `LightBase`, `Sun`, `LightDescription`, `ScatterSky`
- `brightnessA` : `LightAnimData`
- `brightnessKeys` : `LightAnimData`
- `brightnessPeriod` : `LightAnimData`
- `brightnessSmooth` : `LightAnimData`
- `brightnessZ` : `LightAnimData`
- `broken` : `PxMultiActor`
- `bubbleEmitTime` : `PlayerData`
- `buildVisibleTree` : `GuiTreeViewCtrl`
- `bumpAtlas` : `Material`
- `bumpTex` : `Material`
- `buoyancyDensity` : `PhysicsDebrisData`, `PhysicsShapeData`, `PxMultiActorData`
- `buttonMargin` : `GuiIconButtonCtrl`
- `buttonType` : `GuiButtonBaseCtrl`
- c -
• callbackOnA: GuiGameListMenuCtrl
• callbackOnB: GuiGameListMenuCtrl
• callbackOnX: GuiGameListMenuCtrl
• callbackOnY: GuiGameListMenuCtrl
• cameraDecay: RigidShapeData, VehicleData
• cameraDefaultFov: ShapeBaseData
• cameraLag: RigidShapeData, VehicleData
• cameraMaxDist: ShapeBaseData
• cameraMaxFov: ShapeBaseData
• cameraMinDist: ShapeBaseData
• cameraMinFov: ShapeBaseData
• cameraOffset: VehicleData, RigidShapeData, TurretShapeData
• cameraRoll: RigidShapeData, VehicleData
• cameraSpeed: GuiObjectView
• cameraZRot: GuiTSCtrl
• camShakeAmp: ExplosionData, ShapeBaseImageData
• camShakeDuration: ExplosionData
• camShakeFalloff: ExplosionData
• camShakeFreq: ExplosionData, ShapeBaseImageData
• camShakeRadius: ExplosionData
• cancelRename: GuiTreeViewCtrl
• canClose: GuiWindowCtrl
• canCollapse: GuiWindowCtrl
• canKeyFocus: GuiControlProfile
• canMaximize: GuiWindowCtrl
• canMinimize: GuiWindowCtrl
• canMove: GuiWindowCtrl
• canRenameObjects: GuiTreeViewCtrl
• canSave: SimObject
• canSaveDynamicFields: SimObject
• canvasClearColor: LevelInfo
• caption: GuiSeparatorCtrl, GuiRolloutCtrl, GuiPaneControl
• captionID: GuiPaneControl
• caseSensitive: ArrayObject
• casing: ShapeBaseImageData
• castShadows: LightBase, Material, LightDescription, PhysicsDebrisData, ScatterSky, Sun, TerrainBlock
• category: GameBaseData, GuiControlProfile
<table>
<thead>
<tr>
<th>Cell Index</th>
<th>Material</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell Layout</td>
<td>Material</td>
</tr>
<tr>
<td>Cell Size</td>
<td>Material</td>
</tr>
<tr>
<td>Center</td>
<td>GuiSpeedometerHud</td>
</tr>
<tr>
<td>centerY</td>
<td>GuiGraphCtrl</td>
</tr>
<tr>
<td>chanceToHitTarget</td>
<td>Lightning</td>
</tr>
<tr>
<td>changeChildPosition</td>
<td>GuiStackControl</td>
</tr>
<tr>
<td>changeChildSizeToFit</td>
<td>GuiStackControl</td>
</tr>
<tr>
<td>changePath</td>
<td>FileDialog</td>
</tr>
<tr>
<td>channel</td>
<td>SFXParameter</td>
</tr>
<tr>
<td>childBorder</td>
<td>GuiAutoScrollCtrl</td>
</tr>
<tr>
<td>childMargin</td>
<td>GuiScrollCtrl</td>
</tr>
<tr>
<td>clarity</td>
<td>WaterObject</td>
</tr>
<tr>
<td>class</td>
<td>SimObject</td>
</tr>
<tr>
<td>className</td>
<td>SimObject</td>
</tr>
<tr>
<td>clear</td>
<td>GuiPopUpMenuCtrl</td>
</tr>
<tr>
<td>clearAllOnSingleSelection</td>
<td>GuiTreeViewCtrl</td>
</tr>
<tr>
<td>clickCollapse</td>
<td>GuiRolloutCtrl</td>
</tr>
<tr>
<td>clientOnly</td>
<td>PxMultiActorData</td>
</tr>
<tr>
<td>clientSide</td>
<td>TriggerData</td>
</tr>
<tr>
<td>clipColumnText</td>
<td>GuiTextListCtrl</td>
</tr>
<tr>
<td>clippingAngle</td>
<td>DecalData</td>
</tr>
<tr>
<td>cloakable</td>
<td>ShapeBaseImageData</td>
</tr>
<tr>
<td>closeCommand</td>
<td>GuiWindowCtrl</td>
</tr>
<tr>
<td>clumpExponent</td>
<td>GroundCover</td>
</tr>
<tr>
<td>clumpRadius</td>
<td>GroundCover</td>
</tr>
<tr>
<td>colCount</td>
<td>GuiControlArrayControl</td>
</tr>
<tr>
<td></td>
<td>GuiDynamicCtrlArrayControl</td>
</tr>
<tr>
<td>collapsable</td>
<td>GuiPaneControl</td>
</tr>
<tr>
<td>collDamageMultiplier</td>
<td>VehicleData</td>
</tr>
<tr>
<td>collDamageThresholdVel</td>
<td>VehicleData</td>
</tr>
<tr>
<td>collidable</td>
<td>ForestItemData</td>
</tr>
<tr>
<td>collisionTol</td>
<td>VehicleData, RigidShapeData</td>
</tr>
<tr>
<td>collisionType</td>
<td>fxShapeReplicatedStatic, TSStatic</td>
</tr>
<tr>
<td>color</td>
<td>GuiSwatchButtonCtrl, GuiSpeedometerHud, LightBase, LightDescription, Lightning, Sun</td>
</tr>
<tr>
<td>colorA</td>
<td>LightAnimData</td>
</tr>
<tr>
<td>colorArg1</td>
<td>GFXSamplerStateData</td>
</tr>
<tr>
<td>colorArg2</td>
<td>GFXSamplerStateData</td>
</tr>
</tbody>
</table>
- colorArg3: GFXSamplerStateData
- colorBullet: GuiListBoxCtrl
- colorKeys: LightAnimData
- colorMultiply: Material
- colorPeriod: LightAnimData
- colors: SplashData, ParticleData
- colorSmooth: LightAnimData
- colorWriteAlpha: GFXStateBlockData
- colorWriteBlue: GFXStateBlockData
- colorWriteDefined: GFXStateBlockData
- colorWriteGreen: GFXStateBlockData
- colorWriteRed: GFXStateBlockData
- colorZ: LightAnimData
- colSize: GuiDynamicCtrlArrayControl
- colSizes: GuiControlArrayControl
- colSpacing: GuiControlArrayControl, GuiDynamicCtrlArrayControl
- columns: GuiFrameSetCtrl, GuiTextListCtrl
- columnSplit: GuiGameListOptionsProfile
- command: GuiControl
- compareToObjectID: GuiTreeViewCtrl
- computeCRC: ShapeBaseImageData, ShapeBaseData
- coneInsideAngle: SFXEmitter, SFXDescription
- coneOutsideAngle: SFXDescription, SFXEmitter
- coneOutsideVolume: SFXDescription, SFXEmitter
- constantAcceleration: ParticleData
- constantThumbHeight: GuiScrollCtrl
- contactTol: RigidShapeData, VehicleData
- controlMode: Camera
- cookie: Sun, LightDescription, LightBase, ScatterSky
- copyEffect: RenderFormatToken
- coronaEnabled: Sun
- coronaMaterial: Sun
- coronaScale: Sun
- coronaTint: Sun
- coronaUseLightColor: Sun
- correctMuzzleVector: ShapeBaseImageData
- correctMuzzleVectorTP: ShapeBaseImageData
- coverage: CloudLayer
- createHoverHeight : FlyingVehicleData
- createNew : TerrainBlock
- crouchBoundingBox : PlayerData
- crouchForce : PlayerData
- crouchTrigger : Player
- cubeFace : CubemapData
- cubemap : Material, WaterObject
- cubeReflectorDesc : ShapeBaseData
- cullDefined : GFXStateBlockData
- cullMode : GFXStateBlockData
- CullResolution : fxFoliageReplicator
- cursorColor : GuiControlProfile
- customFootstepSound : Material
- customImpactSound : Material
- d -
- damageEmitter: VehicleData
- damageEmitterOffset: VehicleData
- damageFillColor: GuiHealthBarHud, GuiCrossHairHud
- damageFrameColor: GuiCrossHairHud
- damageLevelTolerance: VehicleData
- damageOffset: GuiCrossHairHud
- damageRect: GuiCrossHairHud
- damping: PxCloth, WheeledVehicleSpring
- dampingCoefficient: PxCloth, ForestItemData
- dataBlock: GameBase
- dataFile: Forest
- dayLength: TimeOfDay
- dayScale: TimeOfDay
- Debris: ShapeBaseData, ExplosionData, PhysicsShapeData
- debrisNum: ExplosionData
- debrisNumVariance: ExplosionData
- debrisPhiMax: ExplosionData
- debrisPhiMin: ExplosionData
- debrisShapeName: ShapeBaseData
- debrisThetaMax: ExplosionData
- debrisThetaMin: ExplosionData
- debrisVelocity: ExplosionData
- debrisVelocityVariance: ExplosionData
- DebugBoxHeight: fxFoliageReplicator
- debugRender: GuiGameListMenuCtrl, TerrainBlock, RenderOcclusionMgr, PxMultiActor
- decal: ProjectileData
- decalBias: LevelInfo
- DecalData: PlayerData
- decalOffset: PlayerData
- decalType: fxShapeReplicatedStatic, TSStatic
- decayHFRatio: SFXEnvironment
- decayLFRatio: SFXEnvironment
- decayTime: SFXEnvironment
- defaultFile: FileDialog
- defaultHeight: GuiRolloutCtrl
- defaultPage: GuiTabBookCtrl
- defaultPath: FileDialog
- defaultValue: SFXParameter
defines: **ShaderData**
delayMS: **ExplosionData , SplashData**
delayTimeIn: **SFXPlayList**
delayTimeInVariance: **SFXPlayList**
delayTimeOut: **SFXPlayList**
delayTimeOutVariance: **SFXPlayList**
delayVariance: **ExplosionData , SplashData**
deleteObjectAllowed: **GuiTreeViewCtrl**
deleteOnMouseUp: **GuiDragAndDropControl**
deniedSound: **GuiTextEditCtrl , GuiMLTextCtrl**
density: **SFXEnvironment , ShapeBaseData , PxCloth , WaterObject**
depthFormat: **RenderFormatToken**
depthGradientMax: **WaterObject**
depthGradientTex: **WaterObject**
description: **SFXTrack , SFXSource , SFXParameter**
destroyedLevel: **ShapeBaseData**
destroyedShape: **PhysicsShapeData**
destroyTreeOnSleep: **GuiTreeViewCtrl**
detailAdjust: **ReflectorDesc**
detailAmp: **ForestItemData**
detailDistance: **TerrainMaterial**
detailFreq: **ForestItemData**
detailMap: **TerrainMaterial , Material**
detailNormalMap: **Material**
detailNormalMapStrength: **Material**
detailScale: **Material**
detailSize: **TerrainMaterial**
detailStrength: **TerrainMaterial**
detailTex: **Material**
diffuseColor: **Material**
diffuseMap: **TerrainMaterial , Material**
diffuseSize: **TerrainMaterial**
diffusion: **SFXEnvironment**
disabledLevel: **ShapeBaseData**
disableImposters: **Forest**
disableMove: **Vehicle**
discardAll: **DecalRoad**
displayEnergy: **GuiHealthBarHud**
- dissolveRadius : GroundCover
- distanceFade : GuiShapeNameHud
- distortEndDist : WaterObject
- distortFullDepth : WaterObject
- distortStartDist : WaterObject
- docking : GuiContainer
- doCollision : Precipitation
- done : GuiFadeinBitmapCtrl
- dopplerFactor : SFXAmbience
- doubleSided : Material
- downJetEmitter : FlyingVehicleData
- drag : Camera, ShapeBaseData
- dragCoefficient : ParticleData
- dragForce : RigidShapeData, HoverVehicleData
- dragToItemAllowed : GuiTreeViewCtrl
- drawBottom : SkyBox
- drawBounds : Forest
- drawCells : Forest
- dropAnimateMS : Precipitation
- dropShader : PrecipitationData
- dropSize : Precipitation
- dropsPerSide : PrecipitationData
- dropTexture : PrecipitationData
- dumpInstances : Material
- dumpModel : PxMultiActorData
- dustEmitter : RigidShapeData, PlayerData, VehicleData
- dustHeight : VehicleData, RigidShapeData
- dustTrailEmitter : RigidShapeData, HoverVehicleData
- dustTrailFreqMod : HoverVehicleData
- dustTrailOffset : HoverVehicleData
- DXPixelShaderFile : ShaderData
- DXVertexShaderFile : ShaderData
- dynamic : CubemapData
- dynamicCubemap : Material
- dynamicFarDist : CubemapData
- dynamicFriction : PxMaterial
- dynamicNearDist : CubemapData
- dynamicNonStackExtent : GuiStackControl
- dynamicObjectTypeMask : CubemapData
- **dynamicPos**: `GuiStackControl`
- **dynamicSize**: `GuiStackControl`, `GuiDynamicCtrlArrayControl`, `CubemapData`
- **dynamicType**: `StaticShapeData`
- All
- Functions
- Variables

- a
- b
- c
- d
- e
- f
- g
- h
- i
- j
- k
- l
- m
- n
- o
- p
- q
- r
- s
- t
- u
- v
- w
- z
- echoDepth: SFXEnvironment
- echoTime: SFXEnvironment
- edge: SFXSpace, Zone, OcclusionVolume
- edgeSnap: GuiWindowCtrl
- EditorOpen: MeshRoad, River, DecalRoad
- effectColor: Material
- ejectionAngle: SplashData
- ejectionFreq: SplashData
- ejectionOffset: ParticleEmitterData
- ejectionPeriodMS: ParticleEmitterData
- ejectionVelocity: ParticleEmitterData
- elasticity: DebrisData, ItemData
- elementDist: LightFlareData
- elementRect: LightFlareData
- elementRotate: LightFlareData
- elementScale: LightFlareData
- elementTint: LightFlareData
- elementUseLightColor: LightFlareData
- elevation: ScatterSky, Sun
- elevationMax: ForestBrushElement
- elevationMin: ForestBrushElement
- emap: ShapeBaseImageData
- emissive: WaterObject, Material
- emitter: SplashData, ParticleEmitterNode, ExplosionData
- emitters: DebrisData
- enabled: RenderPassStateToken
- engineBrake: WheeledVehicleData
- engineSound: FlyingVehicleData, HoverVehicleData, WheeledVehicleData
- engineTorque: WheeledVehicleData
- enterCommand: Trigger
- envDiffusion: SFXEnvironment
- environment: SFXAmbience
- envMap: Material
- envSize: SFXEnvironment
- envTex: Material
- escapeCommand: GuiMLTextEditCtrl, GuiTextEditCtrl
- excludedStates: SFXState
- exitingWater: RigidShapeData, PlayerData, VehicleData
- exitSplashSoundVelocity: **VehicleData**, **PlayerData**, **RigidShapeData**
- expanded: **GuiRolloutCtrl**
- explodeOnMaxBounce: **DebrisData**
- Explosion: **SplashData**, **PhysicsShapeData**, **ShapeBaseData**, **DebrisData**, **ProjectileData**
- explosionOffset: **ProximityMineData**
- explosionScale: **ExplosionData**
- explosionShape: **ExplosionData**
- exposure: **ScatterSky**, **CloudLayer**
- extent: **GuiControl**
- eyeOffset: **ShapeBaseImageData**
- eyeRotation: **ShapeBaseImageData**
- All
- Functions
- Variables

- a
- b
- c
- d
- e
- f
- g
- h
- i
- j
- k
- l
- m
- n
- o
- p
- q
- r
- s
- t
- u
- v
- w
- z

- f -
- faceViewer: ExplosionData
- fade: DebrisData
- fadeColor: GuiFadeinBitmapCtrl, Lightning
- fadeDelay: ProjectileData
- fadeDist: Precipitation
- fadeDistEnd: Precipitation
- fadeEndPixelSize: DecalData
- fadeInEase: SFXDescription, GuiFadeinBitmapCtrl
- FadeInRegion: fxFoliageReplicator
- fadeInTime: SFXDescription, SFXEmitter, GuiFadeinBitmapCtrl
- fadeLoops: SFXDescription
- fadeOutEase: SFXDescription, GuiFadeinBitmapCtrl
- FadeOutRegion: fxFoliageReplicator
- fadeOutTime: SFXDescription, SFXEmitter, GuiFadeinBitmapCtrl
- fadeStartDistance: LightBase, LightDescription, ScatterSky, Sun
- fadeStartPixelSize: DecalData
- fadeTime: DecalData
- fadeTimeIn: SFXPlayList
- fadeTimeInVariance: SFXPlayList
- fadeTimeOut: SFXPlayList
- fadeTimeOutVariance: SFXPlayList
- fallback: CustomMaterial
- fallingSpeedThreshold: PlayerData
- farDist: ReflectorDesc
- ffLighting: GFXStateBlockData
- fileFilter: GuiDirectoryFileListCtrl
- fileMustExist: OpenFolderDialog
- fileName: OpenFileDialog, SFXFMODProject, Prefab, SFXEmitter, SFXProfile
- filePath: GuiDirectoryFileListCtrl
- fillColor: GuiHealthBarHud, GuiHealthTextHud, GuiShapeNameHud, GuiControlProfile, GuiClockHud
- fillColorHL: GuiControlProfile
- fillColorNA: GuiControlProfile
- fillColorSEL: GuiControlProfile
- fillRowFirst: GuiDynamicCtrlArrayControl
- filters: `FileDialog`
- firstPerson: `ShapeBaseImageData`
- firstPersonOnly: `ShapeBaseData`
- firstPersonShadows: `PlayerData`
- fitBook: `GuiTabPageCtrl`
- fitParentWidth: ` GuiListBoxCtrl, GuiTextListCtrl`
- FixAspectRatio: `fxFoliageReplicator`
- fixedPanel: `GuiSplitContainer`
- fixedSize: `GuiSplitContainer`
- FixSizeToMax: `fxFoliageReplicator`
- flags: `SFXEnvironment`
- flareEnabled: `LightFlareData`
- flareScale: `LightBase, LightDescription, ScatterSky, Sun`
- flareTexture: `LightFlareData`
- flareType: `LightBase, LightDescription, ScatterSky, Sun`
- flightCeiling: `MissionArea`
- flightCeilingRange: `MissionArea`
- floatingGravMag: `HoverVehicleData`
- floatingThrustFactor: `HoverVehicleData`
- floatSound: `HoverVehicleData`
- FlowMagnitude: `River`
- flush: `Material`
- fmodGroup: `SFXFMODEvent, SFXFMODEventGroup`
- fmodName: `SFXFMODEvent, SFXFMODEventGroup`
- fmodParameterRanges: `SFXFMODEvent`
- fmodParameterValues: `SFXFMODEvent`
- fmodProject: `SFXFMODEventGroup`
- foamAmbientLerp: `WaterObject`
- foamDir: `WaterObject`
- foamMaxDepth: `WaterObject`
- foamOpacity: `WaterObject`
- foamRippleInfluence: `WaterObject`
- foamSpeed: `WaterObject`
- foamTex: `WaterObject`
- foamTexScale: `WaterObject`
- focusOnMouseWheel: `GuiTextEditSliderBitmapCtrl, GuiTextEditSliderCtrl`
- fogAtmosphereHeight: `LevelInfo`
- fogBandHeight: `SkyBox`
- fogColor : *LevelInfo*
- fogDensity : *LevelInfo*
- fogDensityOffset : *LevelInfo*
- fogScale : *ScatterSky*
- FoliageCount : *fxFoliageReplicator*
- FoliageFile : *fxFoliageReplicator*
- FoliageRetries : *fxFoliageReplicator*
- followCam : *Precipitation*
- fontCharset : *GuiControlProfile*
- fontColor : *GuiControlProfile*
- fontColorHL : *GuiControlProfile*
- fontColorLink : *GuiControlProfile*
- fontColorLinkHL : *GuiControlProfile*
- fontColorNA : *GuiControlProfile*
- fontColors : *GuiControlProfile*
- fontColorSEL : *GuiControlProfile*
- fontSize : *GuiControlProfile*
- fontType : *GuiControlProfile*
- FootBubblesSound : *PlayerData*
- FootHardSound : *PlayerData*
- FootMetalSound : *PlayerData*
- footPuffEmitter : *PlayerData*
- footPuffNumParts : *PlayerData*
- footPuffRadius : *PlayerData*
- FootShallowSound : *PlayerData*
- FootSnowSound : *PlayerData*
- FootSoftSound : *PlayerData*
- footstepSoundId : *Material*
- footstepSplashHeight : *PlayerData*
- FootUnderwaterSound : *PlayerData*
- FootWadingSound : *PlayerData*
- force : *WheeledVehicleSpring, Camera*
- forceClose : *GuiPopUpMenuCtrl*
- forceDetail : *TSStatic, fxShapeReplicatedStatic*
- forceFOV : *GuiTSCtrl*
- forceImposters : *Forest*
- forceOnAction : *GuiPopUpMenuCtrl*
- forceUpdateMaterials : *TSShapeConstructor*
- ForestItemData : *ForestBrushElement*
- format: GuiTextEditSliderBitmapCtrl, RenderFormatToken, GuiTextEditSliderCtrl
- forwardJetEmitter: HoverVehicleData, FlyingVehicleData
- forwardLit: CustomMaterial
- frame: DecalData
- frameColor: GuiHealthBarHud, GuiShapeNameHud, GuiClockHud, GuiHealthTextHud
- framesPerSec: ParticleData
- fresnelBias: WaterObject
- fresnelPower: WaterObject
- friction: PhysicsShapeData, ItemData, DebrisData, PhysicsDebrisData, PxCloth
- frontSidePassable: Portal
- frontTabPadding: GuiTabBookCtrl
- frozen: GuiDynamicCtrlArrayControl
- fudgeFactor: GuiFrameSetCtrl
- fullReflect: WaterObject
- fullRowSelect: GuiTreeViewCtrl
- g -
- getAddress : AIConnection
- getAimLocation : AIClient
- getAnimFlags : Material
- getColorById : GuiPopUpMenuCtrlEx
- getFilename : Material
- getFirstRootItem : GuiTreeViewCtrl
- getFreeLook : AIConnection
- getItemCount : GuiTreeViewCtrl
- getLocation : AIConnection
- getMoveDestination : AIConnection
- getSelected : GuiPopUpMenuCtrl
- getSelectedItemList : GuiTreeViewCtrl
- getSelectedItemsCount : GuiTreeViewCtrl
- getTargetObject : AIConnection
- getText : GuiPopUpMenuCtrl
- getValue : GuiControl
- glow : Material
- glowIntensity : Precipitation
- gravityCoefficient : ParticleData
- gravityMod : ItemData, PhysicalZone, ProjectileData
- gravModifier : DebrisData
- gridElementSize : WaterBlock, WaterPlane
- gridSize : GroundCover, WaterBlock, WaterPlane
- GroundAlpha : fxFoliageReplicator
- groundImpactMinSpeed : PlayerData
- groundImpactShakeAmp : PlayerData
- groundImpactShakeDuration : PlayerData
- groundImpactShakeFalloff : PlayerData
- groundImpactShakeFreq : PlayerData
- groupNum : GuiButtonBaseCtrl
- gustFrequency : ForestWindEmitter
- gustStrength : ForestWindEmitter
- gustWobbleStrength : ForestWindEmitter
- gustYawAngle : ForestWindEmitter
- gustYawFrequency : ForestWindEmitter
- gyroDrag : HoverVehicleData
- h -
- hardImpactSound: RigidShapeData, VehicleData
- hardImpactSpeed: VehicleData, RigidShapeData
- hardSplashSoundVelocity: RigidShapeData, VehicleData, PlayerData
- hasBitmapArray: GuiControlProfile
- hasMount: ForestWindEmitter
- headingRate: TurretShapeData
- height: SplashData, BasicClouds, CloudLayer
- HFReference: SFXEnvironment
- hidden: SimObject
- HideFoliage: fxFoliageReplicator
- hideHeader: GuiRolloutCtrl
- HideReplications: fxShapeReplicator
- highResOnly: ParticleEmitterData
- historySize: GuiTextEditCtrl
- hitAreaLowerRight: GuiGameListMenuProfile
- hitAreaUpperLeft: GuiGameListMenuProfile
- hitPlayers: Precipitation
- hitVehicles: Precipitation
- horizMaxSpeed: PlayerData
- horizontalSurfaceForce: FlyingVehicleData
- horizResistFactor: PlayerData
- horizResistSpeed: PlayerData
- horizSizing: GuiControl
- horizStacking: GuiStackControl
- hotSpot: GuiCursor
- hotTrackCallback: GuiPopUpMenuCtrlEx
- hoverHeight: FlyingVehicleData
- hovertime: GuiControl
- hScrollBar: GuiScrollCtrl
- All
- Functions
- Variables

- a
- b
- c
- d
- e
- f
- g
- h
- i
- j
- k
- l
- m
- n
- o
- p
- q
- r
- s
- t
- u
- v
- w
- z

- i -
- iconBitmap: GuiIconButtonCtrl
- iconLocation: GuiIconButtonCtrl
- iconOffset: GuiGameListMenuProfile
- ignoreNodeScale: TSShapeConstructor
- ignoreWater: DebrisData
- imageAnimPrefix: PlayerData, ShapeBaseImageData
- imageAnimPrefixFP: PlayerData, ShapeBaseImageData
- imageTrigger0: Player
- imageTrigger1: Player
- impactForce: ProjectileData
- impactHardSound: PlayerData
- impactMetalSound: PlayerData
- impactSnowSound: PlayerData
- impactSoftSound: PlayerData
- impactSoundId: Material
- impactWaterEasy: RigidShapeData, VehicleData, PlayerData
- impactWaterHard: RigidShapeData, VehicleData, PlayerData
- impactWaterMedium: RigidShapeData, VehicleData, PlayerData
- includedStates: SFXState
- includeLightmappedGeometryInShadow: LightBase, LightDescription, ScatterSky, Sun
- increment: GuiTextEditSliderBitmapCtrl, GuiTextEditSliderCtrl
- indoorWeight: SpawnSphere
- inheritedVelFactor: ParticleData
- inheritEnergyFromMount: ShapeBaseData
- initialPosition: Projectile
- initialVelocity: Projectile
- innerAngle: SpotLight
- InnerRadiusX: fxFoliageReplicator, fxShapeReplicator
- InnerRadiusY: fxFoliageReplicator, fxShapeReplicator
- integration: RigidShapeData, VehicleData
- Interactions: fxShapeReplicator
- interiorFile: InteriorInstance
- internalName: SimObject
- invertLayer: GroundCover
- invisible: GuiSeparatorCtrl
- is3D: SFXDescription, SFXEmitter
- isActive: GuiControl
- isAIControlled: ShapeBase
- isAutoGenerated: Material
- isBallistic: ProjectileData
- isContainer: GuiControl
- isEnabled: LightBase, PostEffect
- isInvincible: ShapeBaseData
- isLooping: Path, SFXDescription, SFXEmitter, GuiAutoScrollCtrl
- isRenderable: River, ScatterSky, SceneObject, SkyBox, Sun, ConvexShape, TimeOfDay, Forest, GameBase, ForestWindEmitter, InteriorInstance, ShapeBase, TerrainBlock, MissionMarker, WayPoint, OcclusionVolume, LightBase, PointLight, StaticShape, SpotLight, DecalManager, RenderObjectExample, RenderShapeExample, MeshRoad, Explosion, fxFoliageReplicator, fxShapeReplicator, AITurretShape, fxShapeReplicatedStatic, GroundCover, ParticleEmitter, PhysicsShape, Splash, PhysicsDebris, PxCloth, SFXSpace, Vehicle, HoverVehicle, BasicClouds, RigidShape, CloudLayer, WaterObject, WaterBlock, Portal, WaterPlane, Marker, Player, SpawnSphere, AIPlayer, Camera, Debris, GroundPlane, Item, PathCamera, PhysicalZone, Zone, Prefab, Projectile, ProximityMine, Trigger, TSStatic, RenderMeshExample, Lightning, ParticleEmitterNode, Precipitation, PhysicsForce, PxMultiActor, SFXEmitter, TurretShape, FlyingVehicle, WheeledVehicle, DecalRoad
- isRenderEnabled: SceneObject
- isSelectable: Zone, DecalRoad, ForestWindEmitter, WheeledVehicle, Prefab, OcclusionVolume, Portal, GameBase, MissionMarker, ShapeBase, LightBase, Sun, Precipitation, WayPoint, Marker, TimeOfDay, SpawnSphere, fxFoliageReplicator, Debris, SFXEmitter, TurretShape, RenderObjectExample, PhysicsShape, ParticleEmitterNode, SFXSpace, Vehicle, RigidShape, InteriorInstance, RenderShapeExample, Projectile, TerrainBlock, Lightning, River, Player, ParticleEmitter, RenderMeshExample, GroundCover, PxCloth, WaterPlane, DecalManager, PointLight, PhysicalZone, ConvexShape, fxShapeReplicator, SpotLight, ProximityMine, AIPlayer, AITurretShape, Item, Forest,
Splash, Trigger, MeshRoad, WaterBlock, WaterObject, BasicClouds, Explosion, ScatterSky, GroundPlane, PathCamera, SceneObject, FlyingVehicle, StaticShape, PxMultiActor, PhysicsDebris, Camera, CloudLayer, HoverVehicle, PhysicsForce, fxShapeReplicatedStatic, AIPlayer, TSStatic, SkyBox

- isSelectionEnabled : SceneObject
- isStreaming : SFXEmitter, SFXDescription
- itemHeight : GuiTreeViewCtrl
- All
- Functions
- Variables

- a
- b
- c
- d
- e
- f
- g
- h
- i
- j
- k
- l
- m
- n
- o
- p
- q
- r
- s
- t
- u
- v
- w
- z

- j -
- jetEnergyDrain: VehicleData
- jetForce: VehicleData
- jetJumpEnergyDrain: PlayerData
- jetJumpForce: PlayerData
- jetJumpSurfaceAngle: PlayerData
- jetMaxJumpSpeed: PlayerData
- jetMinJumpEnergy: PlayerData
- jetMinJumpSpeed: PlayerData
- jetSound: FlyingVehicleData, HoverVehicleData, WheeledVehicleData
- jumpDelay: PlayerData
- jumpEnergyDrain: PlayerData
- jumpForce: PlayerData
- jumpJetTrigger: Player
- jumpSurfaceAngle: PlayerData
- jumpTowardsNormal: PlayerData
- jumpTrigger: Player
- justify: GuiControlProfile
- All
- Functions
- Variables

- a
- b
- c
- d
- e
- f
- g
- h
- i
- j
- k
- l
- m
- n
- o
- p
- q
- r
- s
- t
- u
- v
- w
- z

- k -
- key: ArrayObject
- kineticFriction: WheeledVehicleTire
- All
- Functions
- Variables

- a
- b
- c
- d
- e
- f
- g
- h
- i
- j
- k
- l
- m
- n
- o
- p
- q
- r
- s
- t
- u
- v
- w
- z
- landSequenceTime: PlayerData
- langTableMod: GuiControl
- lastSplitTerrainOnly: LightDescription, ScatterSky, LightBase, Sun
- lateralDamping: WheeledVehicleTire
- lateralForce: WheeledVehicleTire
- lateralRelaxation: WheeledVehicleTire
- layer: GroundCover
- layerEnabled: BasicClouds
- leaveCommand: Trigger
- leftMargin: GuiSeparatorCtrl
- length: GuiSpeedometerHud, WheeledVehicleSpring
- level: ConsoleLogger
- LFReference: SFXEnvironment
- lifeSpan: DecalData
- lifetime: DebrisData, Debris, ProjectileData, PhysicsDebrisData
- lifetimeMS: ExplosionData, SplashData, ParticleData, ParticleEmitterData
- lifetimeVariance: DebrisData, ExplosionData, SplashData, PhysicsDebrisData
- lifetimeVarianceMS: ParticleEmitterData, ParticleData
- lightAmbient: GuiObjectView
- lightBrightness: ShapeBaseImageData
- lightColor: ItemData, GuiObjectView, ShapeBaseImageData
- lightDesc: ProjectileData
- lightDirection: GuiObjectView
- lightDuration: ShapeBaseImageData
- lightEndBrightness: ExplosionData
- lightEndColor: ExplosionData
- lightEndRadius: ExplosionData
- lightMap: Material
- lightMapSize: TerrainBlock
- lightNormalOffset: ExplosionData
- LightOn: fxFoliageReplicator
- lightOnlyStatic: ItemData
- lightRadius: ItemData, ShapeBaseImageData
- lightStartBrightness: ExplosionData
- lightStartColor: ExplosionData
- **lightStartRadius**: `ExplosionData`
- **LightSync**: `fxFoliageReplicator`
- **lightTime**: `ItemData, fxFoliageReplicator`
- **lightType**: `ItemData, ShapeBaseImageData`
- **linearDamping**: `PhysicsShapeData, PhysicsDebrisData`
- **linearDrag**: `PxMultiActorData`
- **linearSleepThreshold**: `PhysicsShapeData, PhysicsDebrisData`
- **lineContinuedIndex**: `GuiMessageVectorCtrl`
- **lineSpacing**: `GuiMLTextCtrl, GuiMessageVectorCtrl`
- **liquidType**: `WaterObject`
- **locked**: `SimObject`
- **lockFrustum**: `GroundCover`
- **lockHorizScroll**: `GuiScrollCtrl`
- **lockMouse**: `GuiMouseEventCtrl`
- **lockVertScroll**: `GuiScrollCtrl`
- **lodReflectScalar**: `Forest`
- **lodType**: `TSShapeConstructor`
- **logWeight**: `LightDescription, LightBase, ScatterSky, Sun`
- **longitudinalDamping**: `WheeledVehicleTire`
- **longitudinalForce**: `WheeledVehicleTire`
- **longitudinalRelaxation**: `WheeledVehicleTire`
- **loopMode**: `SFXPlayList`
- **LowLODDistance**: `River`
- m -
- magFilter: `GFXSamplerStateData`
- mainThrustForce: `HoverVehicleData`
- makeIconSquare: `GuiIconButtonCtrl`
- makeNameCallback: `GuiListBoxCtrl`
- maneuveringForce: `FlyingVehicleData`
- mapTo: `Material`
- margin: `GuiRolloutCtrl`, `GuiContainer`
- markerName: `WayPoint`
- mass: `PhysicsDebrisData`, `ShapeBaseData`, `PhysicsShapeData`, `WheeledVehicleTire`, `ForestItemData`, `Camera`, `ShapeBaseImageData`
- massBox: `RigidShapeData`, `VehicleData`
- massCenter: `RigidShapeData`, `VehicleData`
- matchColor: `GuiMessageVectorCtrl`
- matchVideoSize: `GuiTheoraCtrl`
- Material: `ConvexShape`, `GroundCover`, `PxCloth`, `GroundPlane`, `DecalData`, `RenderMeshExample`, `PxMultiActorData`, `DecalRoad`, `SkyBox`
- matNamePrefix: `TSShapeConstructor`
- maxAngle: `GuiSpeedometerHud`
- maxAnisotropy: `GFXSamplerStateData`
- maxAutoSpeed: `FlyingVehicleData`
- maxBackwardSpeed: `PlayerData`
- maxBillboardTiltAngle: `GroundCover`
- maxChars: `GuiMLTextCtrl`
- maxClumpCount: `GroundCover`
- maxColorIndex: `GuiMessageVectorCtrl`
- maxConcurrentSounds: `ShapeBaseImageData`
- maxCrouchBackwardSpeed: `PlayerData`
- maxCrouchForwardSpeed: `PlayerData`
- maxCrouchSideSpeed: `PlayerData`
- maxDamage: `ShapeBaseData`
- maxDistance: `SFXDescription`, `SFXPlaylist`, `SFXEmitter`
- maxDistanceVariance: `SFXPlaylist`
- maxDrag: `RigidShapeData`, `VehicleData`
- maxElements: `GroundCover`
- maxElevation: `GroundCover`
- maxEnergy: `ShapeBaseData`
- maxForwardSpeed: `PlayerData`
- maxFreelookAngle: PlayerData
- maxHeading: TurretShapeData
- MaxHeight: fxFoliageReplicator
- maxImpulseVelocity: Player
- maxJumpSpeed: PlayerData
- maxLength: GuiTextCtrl
- maxLookAngle: PlayerData
- MaxLuminance: fxFoliageReplicator
- maxMass: Precipitation
- maxOrbitDistance: GuiObjectView
- maxPitch: TurretShapeData
- maxPopupHeight: GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
- maxPredictionTicks: Player
- maxProneBackwardSpeed: PlayerData
- maxProneForwardSpeed: PlayerData
- maxProneSideSpeed: PlayerData
- maxRateMs: ReflectorDesc
- maxScanDistance: ALTurretShapeData
- maxScanHeading: ALTurretShapeData
- maxScanPitch: ALTurretShapeData
- maxSideSpeed: PlayerData
- maxSlope: GroundCover
- maxSpeed: GuiSpeedometerHud, Precipitation
- maxSpinSpeed: DebrisData
- maxSprintBackwardSpeed: PlayerData
- maxSprintForwardSpeed: PlayerData
- maxSprintSideSpeed: PlayerData
- maxSteeringAngle: VehicleData
- maxStepHeight: PlayerData
- MaxSwayTime: fxFoliageReplicator
- maxTimeScale: PlayerData
- maxTurbulence: Precipitation
- maxUnderwaterBackwardSpeed: PlayerData
- maxUnderwaterForwardSpeed: PlayerData
- maxUnderwaterSideSpeed: PlayerData
- maxVelocity: ItemData
- maxWarpTicks: Item, Player
- maxWeaponRange: ALTurretShapeData
- maxWheelSpeed: WheeledVehicleData
- MaxWidth : fxFoliageReplicator
- mediaPath : SFXFMODProject
- mediumSplashSoundVelocity : RigidShapeData, VehicleData, PlayerData
- meshCulling : fxShapeReplicatedStatic, TSStatic
- mieScattering : ScatterSky
- minAngle : GuiSpeedometerHud
- minClumpCount : GroundCover
- minDrag : RigidShapeData, VehicleData
- minElevation : GroundCover
- minEnergy : ShapeBaseImageData
- minExtent : GuiControl
- minFilter : GFXSamplerStateData
- MinHeight : fxFoliageReplicator
- minImpactSpeed : RigidShapeData, VehicleData, PlayerData
- minJetEnergy : VehicleData
- minJumpEnergy : PlayerData
- minJumpSpeed : PlayerData
- minLaterallImpactSpeed : PlayerData
- minLookAngle : PlayerData
- MinLuminance : fxFoliageReplicator
- minMass : Precipitation
- minnaertConstant : Material
- minOrbitDistance : GuiObjectView
- minPitch : TurretShapeData
- minRollSpeed : RigidShapeData, VehicleData
- minRunEnergy : PlayerData
- minSpeed : Precipitation
- minSpinSpeed : DebrisData
- minSprintEnergy : PlayerData
- MinSwayTime : fxFoliageReplicator
- minTabWidth : GuiTabBookCtrl
- minTrailSpeed : FlyingVehicleData
- minWarpTicks : Item, Player
- MinWidth : fxFoliageReplicator
- mipFilter : GFXSamplerStateData
- mipLODBias : GFXSamplerStateData
- mirrorSet : GuiListBoxCtrl
- missionCycleCleanup : AIClient
- mMoveTolerance: AIPlayer
- modal: GuiControlProfile, GuiControl
- modulationDepth: SFXEnvironment
- modulationTime: SFXEnvironment
- moonAzimuth: ScatterSky
- moonElevation: ScatterSky
- moonEnabled: ScatterSky
- moonLightColor: ScatterSky
- moonMat: ScatterSky
- moonScale: ScatterSky
- mountedNode: GuiObjectView
- mountedShapeFile: GuiObjectView
- mountedSkin: GuiObjectView
- mountNode: SceneObject
- mountPID: SceneObject
- mountPoint: ShapeBaseImageData
- mountPos: SceneObject
- mountRot: SceneObject
- mouseDragging: GuiTreeViewCtrl
- mouseOverSelected: GuiControlProfile
- mouseWheelScrollSpeed: GuiScrollCtrl
- move: AIClient
- moveForward: AIClient
- movementSpeed: Camera
- moveStuckTestDelay: AIPlayer
- moveStuckTolerance: AIPlayer
- movingBubblesSound: PlayerData
- msToNext: Marker
- MultipleFiles: OpenFileDialog
- multipleSelections: GuiTreeViewCtrl
- MustExist: OpenFileDialog
- muzzleVelocity: ProjectileData
- n -
- name: SimObject
- nearClip: LevelInfo
- nearDist: ReflectorDesc
- neverImport: TSShapeConstructor
- neverImportMesh: TSShapeConstructor
- newtonMode: Camera
- newtonRotation: Camera
- nightColor: ScatterSky
- nightCubemap: ScatterSky
- nightFogColor: ScatterSky
- nightScale: TimeOfDay
- noBillboards: GroundCover
- noCorrection: PxMultiActorData
- noCorrections: PhysicsShape
- Node: MeshRoad, River, DecalRoad
- noIndividualDamage: StaticShapeData
- normalForce: HoverVehicleData
- normalMap: Material, TerrainMaterial
- noShapes: GroundCover
- noSmoothing: PhysicsShape
- numbersOnly: GuiControlProfile
- numBounces: DebrisData
- numDmgEmitterAreas: VehicleData
- numDrops: Precipitation
- numFences: GuiCanvas
- numSegments: SplashData
- numSlotsToPlay: SFXPlayList
- numSplits: LightDescription, LightBase, Sun, ScatterSky
- All
- Functions
- Variables

- a
- b
- c
- d
- e
- f
- g
- h
- i
- j
- k
- l
- m
- n
- o
- p
- q
- r
- s
- t
- u
- v
- w
- z

- o -
- objectTypeMask: ReflectorDesc
- observeThroughObject: ShapeBaseData
- occlusionRadius: LightFlareData
- offset: ExplosionData, ShapeBaseImageData
- offsetA: LightAnimData
- offsetKeys: LightAnimData
- offsetPeriod: LightAnimData
- offsetSmooth: LightAnimData
- OffsetZ: LightAnimData, fxFoliageReplicator, fxShapeReplicator
- OGLPixelShaderFile: ShaderData
- OGLVertexShaderFile: ShaderData
- onChildAdded: GuiScriptNotifyCtrl
- onChildRemoved: GuiScriptNotifyCtrl
- onChildResized: GuiScriptNotifyCtrl
- oneFrameOnly: PostEffect
- onGainFirstResponder: GuiScriptNotifyCtrl
- onLoseFirstResponder: GuiScriptNotifyCtrl
- onParentResized: GuiScriptNotifyCtrl
- onRenameValidate: GuiTreeViewCtrl
- onResize: GuiScriptNotifyCtrl
- onThisFrame: PostEffect
- opaque: GuiControlProfile
- orbitDistance: GuiObjectView
- orientation: GuiSplitContainer
- orientOnVelocity: ParticleEmitterData
- orientParticles: ParticleEmitterData
- originSort: TSStatic, fxShapeReplicatedStatic
- outdoorWeight: SpawnSphere
- outerAngle: SpotLight
- OuterRadiusX: fxFoliageReplicator, fxShapeReplicator
- OuterRadiusY: fxFoliageReplicator, fxShapeReplicator
- overallFoamOpacity: WaterObject
- overallRippleMagnitude: WaterObject
- overallScale: LightFlareData
- overallWaveMagnitude: WaterObject
- overDarkFactor: LightDescription, Sun, LightBase, ScatterSky
- overlayMap: Material
- overlayTex: Material
- `overrideAdvance`: `ParticleEmitterData`
- `OverwritePrompt`: `SaveFileDialog`

Copyright © GarageGames, LLC. All Rights Reserved.
- p -
- padding: GuiContainer, GuiDynamicCtrlArrayControl, GuiMenuBar, GuiStackControl
- parallaxScale: Material, TerrainMaterial
- parameters: SFXTrack, SFXDescription
- parentGroup: SimObject
- particleDensity: ExplosionData
- ParticleEmitter: ExplosionData, ProjectileData
- particleRadius: ExplosionData
- particles: ParticleEmitterData
- particleWaterEmitter: ProjectileData
- password: GuiTextEditCtrl
- passwordMask: GuiTextEditCtrl
- pauseAnimation: LightBase
- periodVarianceMS: ParticleEmitterData
- persistentId: SimObject
- phiReferenceVel: ParticleEmitterData
- phiVariance: ParticleEmitterData
- physicsPlayerType: PlayerData
- physXStream: PxMultiActorData
- pickupRadius: PlayerData
- pitch: SFXDescription, SFXEmitter
- pitchForce: HoverVehicleData
- pitchRate: TurretShapeData
- pitchScale: SFXPlayList
- pitchScaleVariance: SFXPlayList
- pixelSpecular: Material
- pixVersion: ShaderData
- PlacementAreaHeight: fxFoliageReplicator, fxShapeReplicator
- PlacementColour: fxFoliageReplicator, fxShapeReplicator
- planarReflection: Material
- plane: OcclusionVolume, SFXSpace, Zone
- play: TimeOfDay
- playAmbient: fxShapeReplicatedStatic, TSStatic, PhysicsShape
- playOnAdd: SFXEmitter
- playOnWake: GuiTheoraCtrl
- playSpeed: ExplosionData
- plotColor: GuiGraphCtrl
- plotInterval: GuiGraphCtrl
- plotType: GuiGraphCtrl
- plotVariable: GuiGraphCtrl
- point: Zone, OcclusionVolume, SFXSpace
- polyhedron: Trigger, PhysicalZone
- position: GuiControl, SceneObject
- postApply: SkyBox
- preload: PhysicsDebrisData, SFXProfile
- priority: LightBase, ReflectorDesc, SFXDescription
- probability: ForestBrushElement, GroundCover
- processAddOrder: RenderBinManager
- profile: GuiControl
- profileForChildren: GuiControlProfile
- Projectile: ShapeBaseImageData
- projectileShapeName: ProjectileData
- proneBoundingBox: PlayerData
- proneForce: PlayerData
- proneTrigger: Player
- pulseRate: GuiHealthTextHud, GuiHealthBarHud
- pulseThreshold: GuiHealthTextHud, GuiHealthBarHud

Copyright © GarageGames, LLC. All Rights Reserved.
- q -
• queue : **EventManager**

Copyright © GarageGames, LLC. All Rights Reserved.
Main  Class List  Namespace List  Online

- All
- Functions
- Variables

- a
- b
- c
- d
- e
- f
- g
- h
- i
- j
- k
- l
- m
- n
- o
- p
- q
- r
- s
- t
- u
- v
- w
- z

- r -
- radialEmitter: ForestWindEmitter
- radius: PointLight, WheeledVehicleTire, ForestItemData, GroundCover, SpawnSphere, ForestWindEmitter
- random: SFXPlayList
- RandomFlip: fxFoliageReplicator
- randomize: DecalData
- range: SFXParameter, SpotLight, GuiTextEditSliderBitmapCtrl, GuiTextEditSliderCtrl, LightDescription, GuiSliderCtrl
- rayleighScattering: ScatterSky
- recoverDelay: PlayerData
- recoverRunForceScale: PlayerData
- referenceDistance: SFXDescription, SFXPlayList, SFXEmitter
- referenceDistanceVariance: SFXPlayList
- reflect: Precipitation
- reflectDetailAdjust: WaterObject
- reflections: SFXEnvironment
- reflectionsDelay: SFXEnvironment
- reflectionsPan: SFXEnvironment
- reflectivity: WaterObject
- reflectMaxRateMs: WaterObject
- reflectNormalUp: WaterObject
- reflectPriority: GuiTSCtrl, WaterObject
- reflectScale: GroundCover
- reflectTexSize: WaterObject
- reload: Material, PxMultiActorData
- removeAllChildren: GuiTreeViewCtrl
- removeChildSelectionByValue: GuiTreeViewCtrl
- renameInternal: GuiTreeViewCtrl
- renderBin: PostEffect
- renderCells: GroundCover
- renderCollision: Player
- renderColorInnerCone: SFXEmitter
- renderColorOuterCone: SFXEmitter
- renderColorOutsideVolume: SFXEmitter
- renderColorPlayingInRange: SFXEmitter
- renderColorPlayingOutOfRange: SFXEmitter
- renderColorRangeSphere: SFXEmitter
- renderColorStoppedInRange: SFXEmitter
- renderColorStoppedOutOfRange: SFXEmitter
- `renderDebugInfo`: `GuiTheoraCtrl`
- `renderedBatches`: `GroundCover`
- `renderedBillboards`: `GroundCover`
- `renderedCells`: `GroundCover`
- `renderedShapes`: `GroundCover`
- `renderEmitters`: `SFXEmitter`
- `renderFirstPerson`: `PlayerData`
- `renderMyItems`: `Player`
- `renderMyPlayer`: `Player`
- `renderNormals`: `fxShapeReplicatedStatic`, `TSStatic`
- `renderOffset`: `GuiCursor`
- `renderOrder`: `RenderBinManager`
- `renderPointDistance`: `SFXEmitter`
- `renderPriority`: `DecalData`, `DecalRoad`, `PostEffect`
- `renderRadialIncrements`: `SFXEmitter`
- `renderReflection`: `ParticleEmitterData`
- `renderReflectPass`: `LightFlareData`
- `renderSweepIncrements`: `SFXEmitter`
- `renderTime`: `PostEffect`
- `renderTriggers`: `Trigger`
- `renderWhenDestroyed`: `ShapeBaseData`
- `renderWireframe`: `RenderTerrainMgr`
- `renderZones`: `PhysicalZone`
- `repairRate`: `ShapeBaseData`
- `repeatCount`: `SFXPlayList`
- `replaceText`: `GuiPopUpMenuCtrlEx`
- `replay`: `SFXPlayList`
- `representedInLightmap`: `LightBase`, `LightDescription`, `ScatterSky`, `Sun`
- `resetDelay`: `GuiAutoScrollCtrl`
- `resizeHeight`: `GuiWindowCtrl`
- `resizeWidth`: `GuiWindowCtrl`
- `resolveEffect`: `RenderFormatToken`
- `respawn`: `TurretShape`
- `restitution`: `PhysicsShapeData`, `PxMaterial`, `WheeledVehicleTire`, `PhysicsDebrisData`
- `restorativeForce`: `HoverVehicleData`
- `resultArg`: `GFXSamplerStateData`
- `returnTab`: `GuiControlProfile`
- reverb: SFXEnvironment
- REVERB_CORE0: SFXEnvironment
- REVERB_CORE1: SFXEnvironment
- REVERB_DECAYHFLIMIT: SFXEnvironment
- REVERB_DECAYTIMESCALE: SFXEnvironment
- REVERB_DIRECTHFAUTO: SFXDescription
- REVERB_ECHOTIMESCALE: SFXEnvironment
- REVERB_HIGHQUALITYDPL2REVERB: SFXEnvironment
- REVERB_HIGHQUALITYREVERB: SFXEnvironment
- REVERB_INSTANCE0: SFXDescription
- REVERB_INSTANCE1: SFXDescription
- REVERB_INSTANCE2: SFXDescription
- REVERB_INSTANCE3: SFXDescription
- REVERB_MODULATIONTIMESCALE: SFXEnvironment
- REVERB_REFECTIONSDELYSCALE: SFXEnvironment
- REVERB_REFLECTIONSSCALE: SFXEnvironment
- REVERB_REVERBDELYSCALE: SFXEnvironment
- REVERB_REVERBSCALE: SFXEnvironment
- REVERB_ROOMAUTO: SFXDescription
- REVERB_ROOMHFAUTO: SFXDescription
- reverbAirAbsorptionFactor: SFXDescription
- reverbDelay: SFXEnvironment
- reverbDirect: SFXDescription
- reverbDirectHF: SFXDescription
- reverbDopplerFactor: SFXDescription
- reverbExclusion: SFXDescription
- reverbExclusionLFRatio: SFXDescription
- reverbFlags: SFXDescription
- reverbObstruction: SFXDescription
- reverbObstructionLFRatio: SFXDescription
- reverbOclusion: SFXDescription
- reverbOclusionDirectRatio: SFXDescription
- reverbOclusionLFRatio: SFXDescription
- reverbOclusionRoomRatio: SFXDescription
- reverbOutsideVolumeHF: SFXDescription
- reverbPan: SFXEnvironment
- reverbReverbRolloffFactor: SFXDescription
- reverbRoom: SFXDescription
- reverbRoomHF: SFXDescription
- `reverbRoomRolloffFactor`: `SFXDescription`
- `reverseOrder`: `ParticleEmitterData`
- `reverseTextList`: `GuiPopUpMenuCtrlEx, GuiPopUpMenuCtrl`
- `reverseThrustForce`: `HoverVehicleData`
- `rightPad`: `GuiGameListOptionsProfile`
- `rigidity`: `ForestItemData`
- `ringLifetime`: `SplashData`
- `rippleDir`: `WaterObject`
- `rippleMagnitude`: `WaterObject`
- `rippleSpeed`: `WaterObject`
- `rippleTex`: `WaterObject`
- `rippleTexScale`: `WaterObject`
- `rollForce`: `HoverVehicleData, FlyingVehicleData`
- `rolloffFactor`: `SFXAmbience, SFXDescription`
- `room`: `SFXEnvironment`
- `roomHF`: `SFXEnvironment`
- `roomLF`: `SFXEnvironment`
- `roomRolloffFactor`: `SFXEnvironment`
- `rotA`: `LightAnimData`
- `rotate`: `Item`
- `rotateWithCamVel`: `Precipitation`
- `rotation`: `SceneObject, ShapeBaseImageControl`
- `rotationalDrag`: `FlyingVehicleData`
- `rotationRange`: `ForestBrushElement`
- `rotKeys`: `LightAnimData`
- `rotPeriod`: `LightAnimData`
- `rotPivotOffset`: `Material`
- `rotSmooth`: `LightAnimData`
- `rotSpeed`: `Material`
- `rotZ`: `LightAnimData`
- `rowCount`: `GuiDynamicCtrlArrayControl`
- `rows`: `GuiFrameSetCtrl`
- `rowSize`: `GuiDynamicCtrlArrayControl, GuiControlArrayControl, GuiGameListMenuProfile`
- `rowSpacing`: `GuiDynamicCtrlArrayControl, GuiControlArrayControl`
- `runEnergyDrain`: `PlayerData`
- `runForce`: `PlayerData`
- `runSurfaceAngle`: `PlayerData`
Main	Class List	Namespace List	Online

- All
- Functions
- Variables

- a
- b
- c
- d
- e
- f
- g
- h
- i
- j
- k
- l
- m
- n
- o
- p
- q
- r
- s
- t
- u
- v
- w
- z

- s -
- samplersDefined: GFXStateBlockData
- samplerStates: GFXStateBlockData
- samples: PxCloth
- saveDataFile: Forest
- sbUsesNAColor: GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
- scale: SplashData, ProjectileData, SceneObject
- scaleExponent: ForestBrushElement
- scaleMax: ForestBrushElement
- scaleMin: ForestBrushElement
- scaleU: GroundPlane
- scaleV: GroundPlane
- scanTickFrequency:AITurretShapeData
- scanTickFrequencyVariance:AITurretShapeData
- scatterDistance:SFXDescription, SFXEmitter
- screenError: TerrainBlock
- scriptAnimTransitionTime: ShapeBaseImageData
- scrollDir: Material
- scrollDirection: GuiAutoScrollCtrl
- scrollOutOfSight: GuiAutoScrollCtrl
- scrollSpeed: Material, GuiAutoScrollCtrl
- seed: fxFoliageReplicator, fxShapeReplicator, GroundCover
- SegmentLength: River
- selectedPage: GuiTabBookCtrl
- selfCollision: PxCloth
- separateAlphaBlendDefined: GFXStateBlockData
- separateAlphaBlendDest: GFXStateBlockData
- separateAlphaBlendEnable: GFXStateBlockData
- separateAlphaBlendOp: GFXStateBlockData
- separateAlphaBlendSrc: GFXStateBlockData
- seqNum: Marker
- sequence: TSShapeConstructor
- sequenceFramePerSec: Material
- sequenceSegmentSize: Material
- setAimLocation: AIClient
- setAutoGenerated: Material
- setEnumContent: GuiPopUpMenuCtrlEx
- setFirstResponder: GuiControl
- setMoveDestination: AIClient
- setMoveSpeed: AIClient
- setNoneSelected: GuiPopUpMenuCtrl
- setTargetObject: AIClient
- shader: CustomMaterial, PostEffect
- shadowDarkenColor: LightBase, LightDescription, ScatterSky, Sun
- shadowDistance: LightBase, LightDescription, ScatterSky, Sun
- shadowEnable: ShapeBaseData
- shadowMaxVisibleDistance: ShapeBaseData
- shadowProjectionDistance: ShapeBaseData
- shadowSize: ShapeBaseData
- shadowSoftness: LightDescription, ScatterSky, Sun, LightBase
- shadowSphereAdjust: ShapeBaseData
- shadowType: LightBase, LightDescription, ScatterSky, Sun
- shakeCamera: ExplosionData, ShapeBaseImageData
- ShapeCount: fxShapeReplicator
- shapeCullRadius: GroundCover
- shapeFile: ShapeBaseData, DebrisData, RenderShapeExample, fxShapeReplicator, WheeledVehicleTire, ForestItemData, GuiObjectView, ShapeBaseImageData, PhysicsDebrisData
- shapeFileFP: ShapeBaseImageData
- shapeFilename: GroundCover
- shapeName: fxShapeReplicatedStatic, PhysicsShapeData, TSStatic, PxMultiActorData
- shapeNameFP: PlayerData
- ShapeRetries: fxShapeReplicator
- ShapeRotateMax: fxShapeReplicator
- ShapeRotateMin: fxShapeReplicator
- ShapeScaleMax: fxShapeReplicator
- ShapeScaleMin: fxShapeReplicator
- shapesCastShadows: GroundCover
- shellExitDir: ShapeBaseImageData
- shellExitVariance: ShapeBaseImageData
- shellVelocity: ShapeBaseImageData
- showBatches: DecalRoad, MeshRoad
- showClassNameForUnnamedObjects: GuiTreeViewCtrl
- showClassNames: GuiTreeViewCtrl
- `showDust`: Material
- `showEnergy`: GuiHealthTextHud
- `showFill`: GuiHealthBarHud, GuiHealthTextHud, GuiShapeNameHud, GuiClockHud
- `showFootprints`: Material
- `showFrame`: GuiHealthBarHud, GuiHealthTextHud, GuiShapeNameHud, GuiClockHud
- `showInternalNames`: GuiTreeviewCtrl
- `showNodes`: River
- `showObjectIds`: GuiTreeviewCtrl
- `showObjectNames`: GuiTreeviewCtrl
- `ShowPlacementArea`: fxFoliageReplicator, fxShapeReplicator
- `showRiver`: River
- `showRoad`: DecalRoad, MeshRoad
- `showRoot`: GuiTreeviewCtrl
- `showSpline`: River, DecalRoad, MeshRoad
- `showTrueValue`: GuiHealthTextHud
- `showWalls`: River
- `showWireframe`: River
- `sideMaterial`: MeshRoad
- `simpleServerCollision`: ItemData
- `simType`: PhysicsShapeData
- `singleDetailSize`: TSShapeConstructor
- `singlePlayerOnly`: PxMultiActorData
- `sinkAllKeyEvents`: GuiTextEditCtrl
- `sinkMax`: ForestBrushElement
- `sinkMin`: ForestBrushElement
- `sinkRadius`: ForestBrushElement
- `size`: PxCloth, DecalData, GuiPopUpMenuCtrl, GuiPopUpMenuCtrlEx
- `sizeExponent`: GroundCover
- `sizeIconToButton`: GuiIconButtonCtrl
- `sizeMax`: GroundCover
- `sizeMin`: GroundCover
- `sizes`: ExplosionData, ParticleData
- `skin`: ShapeBase, fxShapeReplicatedStatic, GuiObjectView, TSStatic
- `skip`: PostEffect
- `skyBrightness`: ScatterSky
- slopeMax : ForestBrushElement
- slopeMin : ForestBrushElement
- smoothingType : Marker
- snap : GuiSliderCtrl
- snapOnMaxBounce : DebrisData
- softImpactSound : RigidShapeData, VehicleData
- softImpactSpeed : RigidShapeData, VehicleData
- softnessDistance : ParticleEmitterData
- softSplashSoundVelocity : RigidShapeData, VehicleData
- sort : GuiPopUpMenuCtrl
- sortID : GuiPopUpMenuCtrl
- sortParticles : ParticleEmitterData
- sound : ProjectileData
- soundAmbience : LevelInfo, SFXSpace, WaterObject, Zone
- soundButtonDown : GuiControlProfile
- soundButtonOver : GuiControlProfile
- soundDistanceModel : LevelInfo
- soundProfile : ExplosionData, PrecipitationData, SplashData
- soundTrack : SFXAmbience
- sourceGroup : SFXDescription, SFXEmitter
- sourceObject : Projectile
- sourceSlot : Projectile
- spawnClass : SpawnSphere
- spawnDatablock : SpawnSphere
- spawnProperties : SpawnSphere
- spawnScript : SpawnSphere
- spawnTransform : SpawnSphere
- specular : Material
- specularColor : WaterObject
- specularMap : Material
- specularPower : WaterObject, Material
- speedMultiplier : Camera
- sphereWeight : SpawnSphere
- spinRandomMax : ParticleData
- spinRandomMin : ParticleData
- spinSpeed : ParticleData
- Splash : PlayerData, ProjectileData
- splashAngle : PlayerData
- splashEmitter : VehicleData, RigidShapeData, PlayerData
- `splashesPerSide`: `PrecipitationData`
- `splashFreqMod`: `RigidShapeData`, `VehicleData`, `PlayerData`
- `splashMS`: `Precipitation`
- `splashShader`: `PrecipitationData`
- `splashSize`: `Precipitation`
- `splashTexture`: `PrecipitationData`
- `splashVelEpsilon`: `PlayerData`, `RigidShapeData`, `VehicleData`
- `splashVelocity`: `PlayerData`
- `splitPoint`: `GuiSplitContainer`
- `splitterSize`: `GuiSplitContainer`
- `sprintCanJump`: `PlayerData`
- `sprintEnergyDrain`: `PlayerData`
- `sprintForce`: `PlayerData`
- `sprintPitchScale`: `PlayerData`
- `sprintStrafeScale`: `PlayerData`
- `sprintTrigger`: `Player`
- `sprintYawScale`: `PlayerData`
- `squareSize`: `TerrainBlock`, `GroundPlane`
- `squealSound`: `WheeledVehicleData`
- `stabDampingConstant`: `HoverVehicleData`
- `stabLenMax`: `HoverVehicleData`
- `stabLenMin`: `HoverVehicleData`
- `stabSpringConstant`: `HoverVehicleData`
- `stackingType`: `GuiStackControl`
- `startDelay`: `GuiAutoScrollCtrl`
- `startLoaded`: `TurretShapeData`
- `startRadius`: `SplashData`
- `startTime`: `TimeOfDay`
- `state`: `SFXPlayList`
- `stateAllowImageChange`: `ShapeBaseImageData`
- `stateAlternateFire`: `ShapeBaseImageData`
- `stateBlock`: `PostEffect`, `CustomMaterial`
- `stateDirection`: `AITurretShapeData`, `ShapeBaseImageData`
- `stateEjectShell`: `ShapeBaseImageData`
- `stateEmitter`: `ShapeBaseImageData`
- `stateEmitterNode`: `ShapeBaseImageData`
- `stateEmitterTime`: `ShapeBaseImageData`
- `stateEnergyDrain`: `ShapeBaseImageData`
- `stateFire`: `AITurretShapeData`, `ShapeBaseImageData`
- stateIgnoreLoadedForReady: ShapeBaseImageData
- stateLoadedFlag: ShapeBaseImageData
- stateMode: SFXPlayList
- stateName: ShapeBaseImageData, AITurretShapeData
- stateRecoil: ShapeBaseImageData
- stateReload: ShapeBaseImageData
- states: SFXAmbience
- stateScaleAnimation: AITurretShapeData, ShapeBaseImageData
- stateScaleAnimationFP: ShapeBaseImageData
- stateScaleShapeSequence: ShapeBaseImageData
- stateScan: AITurretShapeData
- stateScript: AITurretShapeData, ShapeBaseImageData
- stateSequence: ShapeBaseImageData, AITurretShapeData
- stateSequenceNeverTransition: ShapeBaseImageData
- stateSequenceRandomFlash: ShapeBaseImageData
- stateSequenceTransitionIn: ShapeBaseImageData
- stateSequenceTransitionOut: ShapeBaseImageData
- stateSequenceTransitionTime: ShapeBaseImageData
- stateShapeSequence: ShapeBaseImageData
- stateSound: ShapeBaseImageData
- stateSpinThread: ShapeBaseImageData
- stateTimeoutValue: ShapeBaseImageData, AITurretShapeData
- stateToken: RenderPassStateBin
- stateTransitionGeneric0In: ShapeBaseImageData
- stateTransitionGeneric0Out: ShapeBaseImageData
- stateTransitionGeneric1In: ShapeBaseImageData
- stateTransitionGeneric1Out: ShapeBaseImageData
- stateTransitionGeneric2In: ShapeBaseImageData
- stateTransitionGeneric2Out: ShapeBaseImageData
- stateTransitionGeneric3In: ShapeBaseImageData
- stateTransitionGeneric3Out: ShapeBaseImageData
- stateTransitionOnActivated: AITurretShapeData
- stateTransitionOnAltTriggerDown: ShapeBaseImageData
- stateTransitionOnAltTriggerUp: ShapeBaseImageData
- stateTransitionOnAmmo: ShapeBaseImageData
- stateTransitionOnAtRest: AITurretShapeData
- stateTransitionOnDeactivated: AITurretShapeData
- stateTransitionOnLoaded: ShapeBaseImageData
- stateTransitionOnMotion: ShapeBaseImageData
- stateTransitionOnNoAmmo: ShapeBaseImageData
- stateTransitionOnNoMotion: ShapeBaseImageData
- stateTransitionOnNoTarget: AITurretShapeData, ShapeBaseImageData
- stateTransitionOnNotAtRest: AITurretShapeData
- stateTransitionOnNotLoaded: ShapeBaseImageData
- stateTransitionOnNotWet: ShapeBaseImageData
- stateTransitionOnTarget: ShapeBaseImageData, AITurretShapeData
- stateTransitionOnTimeout: AITurretShapeData, ShapeBaseImageData
- stateTransitionOnTriggerDown: ShapeBaseImageData
- stateTransitionOnTriggerUp: ShapeBaseImageData
- stateTransitionOnWet: ShapeBaseImageData
- stateWaitForTimeout: ShapeBaseImageData, AITurretShapeData
- static: Item
- staticFriction: PhysicsDebrisData, PxMaterial, WheeledVehicleTire, PhysicsShapeData
- staticOnMaxBounce: DebrisData
- statusCallback: SFXSource
- steeringForce: HoverVehicleData, FlyingVehicleData
- steeringRollForce: FlyingVehicleData
- stencilDefined: GFXStateBlockData
- stencilEnable: GFXStateBlockData
- stencilFailOp: GFXStateBlockData
- stencilFunc: GFXStateBlockData
- stencilMask: GFXStateBlockData
- stencilPassOp: GFXStateBlockData
- stencilRef: GFXStateBlockData
- stencilWriteMask: GFXStateBlockData
- stencilZFailOp: GFXStateBlockData
- sticky: ItemData
- stop: AIClient
- stopOnSleep: GuiTheoraCtrl
- strafeThrustForce: HoverVehicleData
- streamPacketSize: SFXDescription
- streamReadAhead: SFXDescription
- strength: ForestWindEmitter
- strikeRadius: Lightning
- strikeSound: LightningData
- strikesPerMinute: Lightning
- strikeTextures: LightningData
- strikeWidth: Lightning
- string: PxMultiActorData
- SubdivideLength: River
- subExplosion: ExplosionData
- subSurface: Material
- subSurfaceColor: Material
- subSurfaceRolloff: Material
- sunScale: ScatterSky
- superClass: SimObject
- surface: ConvexShape
- SwayMagFront: fxFoliageReplicator
- SwayMagSide: fxFoliageReplicator
- SwayOn: fxFoliageReplicator
- SwaySync: fxFoliageReplicator
- swimBoundingBox: PlayerData
- swimForce: PlayerData
• All
• Functions
• Variables

• a
• b
• c
• d
• e
• f
• g
• h
• i
• j
• k
• l
• m
• n
• o
• p
• q
• r
• s
• t
• u
• v
• w
• z

- t -
- **tab**: `GuiControlProfile`
- **tabComplete**: `GuiTextEditCtrl`
- **tabHeight**: `GuiTabBookCtrl`
- **tabMargin**: `GuiTabBookCtrl`
- **tabPosition**: `GuiTabBookCtrl`
- **tabSize**: `GuiTreeViewCtrl`
- **tail**: `GuiSpeedometerHud`
- **target**: `CustomMaterial, PostEffect`
- **targetClear**: `PostEffect`
- **targetClearColor**: `PostEffect`
- **targetDepthStencil**: `PostEffect`
- **targetFormat**: `PostEffect`
- **targetScale**: `PostEffect`
- **targetSize**: `PostEffect`
- **team**: `WayPoint`
- **terminalVelocity**: `DebrisData`
- **TerrainAlignment**: `fxShapeReplicator`
- **terrainFile**: `TerrainBlock`
- **texCols**: `DecalData`
- **texDirection**: `BasicClouds, CloudLayer`
- **texFactor**: `SplashData`
- **texOffset**: `BasicClouds`
- **texRows**: `DecalData`
- **texScale**: `BasicClouds, CloudLayer`
- **texSize**: `Sun, LightBase, ReflectorDesc, LightDescription, ScatterSky`
- **texSpeed**: `BasicClouds, CloudLayer`
- **text**: `GuiButtonBaseCtrl, GuiWindowCtrl, GuiTextCtrl, GuiMLTextCtrl`
- **textColor**: `GuiHealthTextHud, GuiShapeNameHud, GuiClockHud`
- **textID**: `GuiButtonBaseCtrl, GuiTextCtrl`
- **textLocation**: `GuiIconButtonCtrl`
- **textMargin**: `GuiIconButtonCtrl`
- **textOffset**: `GuiControlProfile, GuiTreeViewCtrl`
- **texture**: `DebrisData, SplashData, BasicClouds, CloudLayer, PostEffect`
- **textureColorOp**: `GFXSamplerStateData`
- **textureCoordCount**: `DecalData`
- textureCoords: DecalData, ParticleData
- textureFactor: GFXStateBlockData
- textureLength: DecalRoad, MeshRoad
- textureName: ParticleData, ParticleEmitterData
- textureTransform: GFXSamplerStateData
- texWrap: SplashData
- theoraFile: GuiTheoraCtrl
- thetaMax: ParticleEmitterData
- thetaMin: ParticleEmitterData
- thickness: PxCloth
- thunderSounds: LightningData
- tickCommand: Trigger
- tickPeriodMS: TriggerData
- ticks: GuiSliderCtrl
- tightnessCoefficient: ForestItemData
- tile: GuiChunkedBitmapCtrl
- time: TimeOfDay
- timeMultiple: ParticleEmitterNodeData
- times: ExplosionData, SplashData, ParticleData
- tireEmitter: WheeledVehicleData
- title: FileDialog
- toneMap: Material
- tooltip: GuiControl
- tooltipOnWidthOnly: GuiTreeViewCtrl
- tooltipProfile: GuiControl
- topMaterial: MeshRoad
- toQueue: MessageForwarder
- trace: SFXController, SFXPlayList
- track: SFXEmitter, SFXPlayList
- trackLostTargetTime: AITurretShapeData
- trailEmitter: FlyingVehicleData
- transcoder: GuiTheoraCtrl
- transitionIn: SFXPlayList
- transitionOut: SFXPlayList
- transitionToLand: PlayerData
- translucent: Material
- translucentBlendOp: Material
- translucentZWrite: Material
- triangleCollision: PxCloth
- triggerDelay: ProximityMineData
- triggerDustHeight: RigidShapeData, VehicleData
- triggerOnOwner: ProximityMineData
- triggerRadius: ProximityMineData
- triggerSound: ProximityMineData
- triggerSpeed: ProximityMineData
- triggerTrailHeight: HoverVehicleData
- trunkBendScale: ForestItemData
- turboFactor: HoverVehicleData
- turbulenceFrequency: ForestWindEmitter
- turbulenceSpeed: Precipitation
- turbulenceStrength: ForestWindEmitter
- type: Marker, GuiSeparatorCtrl
- All
- Functions
- Variables

- a
- b
- c
- d
- e
- f
- g
- h
- i
- j
- k
- l
- m
- n
- o
- p
- q
- r
- s
- t
- u
- v
- w
- z

- u -
• underwaterColor : WaterObject
• underwaterExplosion : ShapeBaseData
• unit : TSShapeConstructor
• upAxis : TSShapeConstructor
• updateDelay : DecalRoad
• upMaxSpeed : PlayerData
• upResistFactor : PlayerData
• upResistSpeed : PlayerData
• useAmbientLightColor : Zone
• useAnisotropic : Material
• UseCulling : fxFoliageReplicator
• useCustomReverb : SFXDescription
• UseDebugInfo : fxFoliageReplicator
• useDevicePixVersion : ShaderData
• useEmitterColors : ParticleEmitterData
• useEmitterSizes : ParticleEmitterData
• useEyeNode : ShapeBaseImageData
• useEyePoint : ShapeBaseData
• useFog : Lightning
• useHardware : SFXDescription
• useInspectorTooltips : GuiTreeViewCtrl
• useInvAlpha : ParticleData
• useLighting : Precipitation
• useModifiers : GuiBitmapButtonCtrl
• useMouseEvents : GuiButtonBaseCtrl
• useNightCubemap : ScatterSky
• useOcclusionQuery : WaterObject, ReflectorDesc
• useRadiusMass : DebrisData
• useRemainderDT : ShapeBaseImageData
• usesEnergy : ShapeBaseImageData
• useSiblingScroller : GuiConsoleEditCtrl
• useSideProjection : TerrainMaterial
• useStates : GuiBitmapButtonCtrl
• useTrackDescriptionOnly : SFXEmitter
• useTrueBillboards : Precipitation, fxFoliageReplicator
• useTurbulence : Precipitation
• useURLMouseCursor : GuiMLTextCtrl
• useVariable : GuiChunkedBitmapCtrl
• useWind : Precipitation
- v -
- validate: GuiTextEditCtrl
- value: GuiSliderCtrl, SFXParameter
- variable: GuiControl
- vehicleDismountTrigger: Player
- velInheritFactor: ProjectileData
- velocity: SplashData, ParticleEmitterNode, DebrisData
- velocityMod: PhysicalZone
- velocityVariance: DebrisData, ParticleEmitterData
- version: CustomMaterial
- vertColor: Material
- vertexColorEnable: GFXStateBlockData
- vertFactor: RigidShapeData, HoverVehicleData
- verticalOffset: GuiShapeNameHud
- verticalSurfaceForce: FlyingVehicleData
- vertLit: Material
- vertSizing: GuiControl
- vertStacking: GuiStackControl
- vertThrustMultiple: FlyingVehicleData
- ViewClosest: fxFoliageReplicator
- ViewDistance: fxFoliageReplicator
- viscosity: WaterObject
- visible: GuiControl
- visibleDistance: LevelInfo
- volume: SFXDescription, SFXEmitter
- volumeScale: SFXPlayList
- volumeScaleVariance: SFXPlayList
- vScrollBar: GuiScrollCtrl
- All
- Functions
- Variables

- a
- b
- c
- d
- e
- f
- g
- h
- i
- j
- k
- l
- m
- n
- o
- p
- q
- r
- s
- t
- u
- v
- w
- z

- w -
- `waitTime`: `GuiFadeinBitmapCtrl`
- `warningColor`: `GuiHealthTextHud`
- `warnThreshold`: `GuiHealthTextHud`
- `waterBreathSound`: `PlayerData`
- `waterDampingScale`: `PhysicsShapeData`, `PhysicsDebrisData`
- `waterDragScale`: `PxMultiActorData`
- `waterExplosion`: `ProjectileData`
- `waterFogDensity`: `WaterObject`
- `waterFogDensityOffset`: `WaterObject`
- `waterWakeSound`: `RigidShapeData`, `VehicleData`
- `waveAmp`: `Material`
- `waveDir`: `WaterObject`
- `waveFreq`: `Material`
- `waveMagnitude`: `WaterObject`
- `waveSpeed`: `WaterObject`
- `waveType`: `Material`
- `weaponLeadVelocity`: `AITurretShapeData`
- `weaponLinkType`: `TurretShapeData`
- `wetDarkening`: `WaterObject`
- `wetDepth`: `WaterObject`
- `WheelImpactSound`: `WheeledVehicleData`
- `width`: `SplashData`, `GuiSpeedometerHud`
- `widthSubdivisions`: `MeshRoad`
- `willFirstRespond`: `GuiScrollCtrl`
- `windCoefficient`: `ParticleData`
- `windDirection`: `GroundCover`
- `windEnabled`: `ForestWindEmitter`
- `windGustFrequency`: `GroundCover`
- `windGustLength`: `GroundCover`
- `windGustStrength`: `GroundCover`
- `windScale`: `GroundCover`, `ForestItemData`
- `windSpeed`: `CloudLayer`
- `windTurbulenceFrequency`: `GroundCover`
- `windTurbulenceStrength`: `GroundCover`
- `wireframe`: `WaterObject`, `MeshRoad`, `DecalRoad`
- `workingQueryBoxSizeMultiplier`: `Vehicle`
- `workingQueryBoxStaleThreshold`: `Vehicle`
- `wrap`: `GuiBitmapCtrl`
Copyright © GarageGames, LLC. All Rights Reserved.
• All
• Functions
• Variables

• a
• b
• c
• d
• e
• f
• g
• h
• i
• j
• k
• l
• m
• n
• o
• p
• q
• r
• s
• t
• u
• v
• w
• z

- z -
- zBias: GFXStateBlockData
- zDefined: GFXStateBlockData
- zEnable: GFXStateBlockData
- zFunc: GFXStateBlockData
- zOffset: GroundCover
- zoneGroup: InteriorInstance, Zone
- zRotOnly: TurretShapeData
- zSlopeBias: GFXStateBlockData
- zWriteEnable: GFXStateBlockData