Common utility classes that are used throughout the whole framework
## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArrayEqualityComparer</td>
<td>ThisEqualityComparer can compare if an array of objects is equal to another</td>
</tr>
<tr>
<td></td>
<td>array of objects by comparing the contained objects itself</td>
</tr>
<tr>
<td>ExtensionMethods</td>
<td>Holds all Extension methods that are used throughout the framework</td>
</tr>
</tbody>
</table>
### Interfaces

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICloneable(T)</td>
<td>Supports cloning, which creates a new instance of the same type with the same value as an existing instance.</td>
</tr>
</tbody>
</table>
ArrayEqualityComparer Class

This EqualityComparer can compare if an array of objects is equal to another array of objects by comparing the contained objects itself.

Namespace: Common
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public class ArrayEqualityComparer : IEqualityComparer</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Class ArrayEqualityComparer _ Implements IEqualityComparer(Of Object())</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public ref class ArrayEqualityComparer : IEqualityComparer</td>
</tr>
</tbody>
</table>

Inheritance Hierarchy

System.Object

Common.ArrayEqualityComparer
See Also

ArrayEqualityComparer Members
Common Namespace
The **ArrayEqualityComparer** type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArrayEqualityComparer</td>
<td>Initializes a new instance of the <code>ArrayEqualityComparer</code> class</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals(Object)</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Equals(Object[], Object[])</strong></td>
<td>Determines whether the specified objects are equal.</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode()</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode(Object[])</strong></td>
<td>Returns a hash code for the specified object.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

ArrayEqualityComparer Class
Common Namespace
Initializes a new instance of the `ArrayEqualityComparer` class

**Namespace:** Common

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>public ArrayEqualityComparer()</code></td>
<td><code>Public Sub New</code></td>
<td><code>public: ArrayEqualityComparer()</code></td>
</tr>
</tbody>
</table>
See Also

ArrayEqualityComparer Class
Common Namespace
The `ArrayEqualityComparer` type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals(Object)</td>
<td>Determines whether the specified Object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>Equals(Object[], Object[])</td>
<td>Determines whether the specified objects are equal.</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode()</td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode(Object[])</td>
<td>Returns a hash code for the specified object.</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a String that represents the current Object. (Inherited from Object.)</td>
</tr>
</tbody>
</table>
See Also

ArrayEqualityComparer Class
Common Namespace
Optimization Framework

ArrayEqualityComparer.Equals Method

ArrayEqualityComparer Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Equals(Object)</code></td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><code>Equals(Object[], Object[])</code></td>
<td>Determines whether the specified objects are equal.</td>
</tr>
</tbody>
</table>
See Also

ArrayEqualityComparer Class
ArrayEqualityComparer Members
Common Namespace
Determines whether the specified objects are equal.

**Namespace:** Common  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
**Syntax**

**C#**

```csharp
public bool Equals(
    Object[] x,
    Object[] y
)
```

**Visual Basic**

```vbnet
Public Function Equals (_
    x As Object(), _
    y As Object() _
) As Boolean
```

**Visual C++**

```c++
public:
    virtual bool Equals(
        array<Object^>^ x,
        array<Object^>^ y
    ) sealed
```

**Parameters**

$x$
Type: `System.Object[]`
The first object of type $T$ to compare.

$y$
Type: `System.Object[]`
The second object of type $T$ to compare.

**Return Value**
true if the specified objects are equal; otherwise, false.

**Implements**

`IEqualityComparer<T>.Equals(T, T)`
See Also

ArrayEqualityComparer Class
Equals Overload
Common Namespace
Optimization Framework

`ArrayEqualityComparer.GetHashCode` Method

`ArrayEqualityComparer` Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetHashCode()</td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode(Object[])</td>
<td>Returns a hash code for the specified object.</td>
</tr>
</tbody>
</table>
See Also

ArrayEqualityComparer Class
ArrayEqualityComparer Members
Common Namespace
Returns a hash code for the specified object.

**Namespace:** Common

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public int GetHashCode($
    Object[] obj
)
```

#### Visual Basic

```vbnet
Public Function GetHashCode ($
    obj As Object()$
) As Integer
```

#### Visual C++

```cpp
public: 
virtual int GetHashCode($
    array<
Object^ $> obj
)
```

#### Parameters

- **obj**
  - Type: `System.Object[]`
  - The `Object` for which a hash code is to be returned.

#### Return Value

A hash code for the specified object.

#### Implements

`IEqualityComparer<T>.GetHashCode(T)`
### Exceptions

<table>
<thead>
<tr>
<th>Exception</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>System.ArgumentNullException</code></td>
<td>The type of <code>obj</code> is a reference type and <code>obj</code> is null.</td>
</tr>
</tbody>
</table>
See Also

ArrayEqualityComparer Class
GetHashCode Overload
Common Namespace
Holds all Extension methods that are used throughout the framework

**Namespace:** Common

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public static class ExtensionMethods
```

### Visual Basic

```vbnet
<ExtensionAttribute> _
Public NotInheritable Class ExtensionMethods
```

### Visual C++

```cpp
[ExtensionAttribute]
public ref class ExtensionMethods abstract sealed
```
Inheritance Hierarchy

System.Object
Common.ExtensionMethods
See Also

ExtensionMethods Members
Common Namespace
The ExtensionMethods type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CartesianProduct(T)</code></td>
<td>Cartesian the product.</td>
</tr>
<tr>
<td><code>Clone(T)</code></td>
<td></td>
</tr>
<tr>
<td><code>ForEach(T)</code></td>
<td>Executes action on every element of source.</td>
</tr>
<tr>
<td><code>MultiplyElements</code></td>
<td>Multiplies the elements and returns the result.</td>
</tr>
</tbody>
</table>
See Also

ExtensionMethods Class
Common Namespace
The **ExtensionMethods** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CartesianProduct(T)</td>
<td>Cartesians the product.</td>
</tr>
<tr>
<td>Clone(T)</td>
<td></td>
</tr>
<tr>
<td>ForEach(T)</td>
<td>Executes action on every element of source</td>
</tr>
<tr>
<td>MultiplyElements</td>
<td>Multiplies the elements and returns the result.</td>
</tr>
</tbody>
</table>
See Also

ExtensionMethods Class
Common Namespace
Cartesians the product.

**Namespace:** Common

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```
public static IEnumerable<IEnumerable<T>> CartesianProduct<T>(
    this IEnumerable<IEnumerable<T>> sequences)
```

**Visual Basic**

```
<ExtensionAttribute>
Public Shared Function CartesianProduct(Of T)(
    sequences As IEnumerable(IEnumerable(Of T)))
    As IEnumerable(IEnumerable(Of T))
```

**Visual C++**

```
[ExtensionAttribute]
public:
    template<typename T>
    static IEnumerable<IEnumerable<T>^>^ CartesianProduct(
        IEnumerable<IEnumerable<T>^>^ sequences)
```

### Parameters

- **sequences**
  - Type: `System.Collections.Generic(IEnumerable(IEnumerable<T>))`
  - The sequences.
**Type Parameters**

*T*  
[Missing <typeparam name="T"/> documentation for  

**Return Value**

[Missing <returns> documentation for  

**Usage Note**

In Visual Basic and C#, you can call this method as an instance method on any object of type `IEnumerable(IEnumerable(T))`. When you use instance method syntax to call this method, omit the first parameter. For more information, see or .
See Also

ExtensionMethods Class
Common Namespace
ExtensionMethods.Clone(T) Method

Namespace: Common
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0

[Missing <summary> documentation for "M:Common.ExtensionMethods.Clone`1`0"]
### Syntax

**C#**

```csharp
public static T Clone<T>(
    this T source
)
```

**Visual Basic**

```vbnet
<ExtensionAttribute>_  
Public Shared Function Clone(Of T) ( _
    source As T _
) As T
```

**Visual C++**

```cpp
[ExtensionAttribute]
public:
    generic<typename T>
    static T Clone(
        T source
    )
```

### Parameters

- **source**
  - Type: `T`
  - [Missing <param name="source"/> documentation for "M:Common.ExtensionMethods.Clone``1(``0)"]
### Type Parameters

\( T \)

[Missing <typeparam name="T"/> documentation for "M:Common.ExtensionMethods.Clone`1(`0)"

### Return Value

[Missing <returns> documentation for "M:Common.ExtensionMethods.Clone`1(`0)"

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type \( T \). When you use instance method syntax to call this method, omit the first parameter. For more information, see or .
See Also

ExtensionMethods Class
Common Namespace
Optimization Framework

ExtensionMethods.ForEach($T$) Method

Namespace: Common
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0

Executes action on every element of source
**Syntax**

### C#

```csharp
public static void ForEach<T>(
    this IEnumerable<T> source,
    Action<T> action
)
```

### Visual Basic

```vbnet
<ExtensionAttribute> _
Public Shared Sub ForEach(Of T) (_
    source As IEnumerable(Of T), _
    action As Action(Of T) _
)
```

### Visual C++

```cpp
[ExtensionAttribute]
public:
generic<typename T>
static void ForEach(
    IEnumerable<T>^ source,
    Action<T>^ action
)
```

**Parameters**

- **source**
  Type: `System.Collections.Generic.IEnumerable<T>()`
  The source.

- **action**
  Type: `System.Action<T>()`
  The action.
Type Parameters

\[ T \]

[Missing <typeparam name="T"/> documentation for "M:Common.ExtensionMethods.ForEach\`1(System.Collections.Generic.IEnumerable\`1{T})"

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IEnumerable(T). When you use instance method syntax to call this method, omit the first parameter. For more information, see or .
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>action != null</td>
</tr>
<tr>
<td>source != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

ExtensionMethods Class
Common Namespace
ExtensionMethods.MultiplyElements Method

Multiplies the elements and returns the result.

**Namespace:** Common

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public static double MultiplyElements(</td>
</tr>
<tr>
<td></td>
<td>this IEnumerable&lt;double&gt; source</td>
</tr>
<tr>
<td></td>
<td>)</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>&lt;ExtensionAttribute&gt; _</td>
</tr>
<tr>
<td></td>
<td>Public Shared Function MultiplyElements ( _</td>
</tr>
<tr>
<td></td>
<td>source As IEnumerable(Of Double) _</td>
</tr>
<tr>
<td></td>
<td>) As Double</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>[ExtensionAttribute]</td>
</tr>
<tr>
<td></td>
<td>public:</td>
</tr>
<tr>
<td></td>
<td>static double MultiplyElements(</td>
</tr>
<tr>
<td></td>
<td>IEnumerable&lt;double&gt;^ source</td>
</tr>
<tr>
<td></td>
<td>)</td>
</tr>
</tbody>
</table>

### Parameters

- **source**
  Type: System.Collections.Generic.IEnumerable<Double>
  The source.

### Return Value

[Missing <returns> documentation for

### Usage Note
In Visual Basic and C#, you can call this method as an instance method on any object of type `IEnumerable<Double>`. When you use instance method syntax to call this method, omit the first parameter. For more information, see or.
## Contracts

### Requires

| source != null |

Learn more about contracts
See Also

ExtensionMethods Class
Common Namespace
ICloneable\((T)\) Interface

Supports cloning, which creates a new instance of the same type with the same value as an existing instance.

**Namespace:** Common

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public interface ICloneable&lt;T&gt; : ICloneable</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Interface ICloneable(Of T) _ Inherits ICloneable</td>
</tr>
</tbody>
</table>
| **Visual C++** | generic<typename T>  
public interface class ICloneable : ICloneable          |
Type Parameters

$T$

The type of the original instance.
See Also

ICloneable(T) Members
Common Namespace
Optimization Framework

ICloneable(T) Members

ICloneable(T) Interface  Methods  See Also  Send Feedback
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="Clone.png" alt="Clone method" /></td>
<td>Creates a new object that is a copy of the current instance. (Inherited from <a href="#">ICloneable</a>.)</td>
</tr>
<tr>
<td><img src="Clone.png" alt="Clone method" /></td>
<td>Creates a new object of type $T$ that is a copy of the current instance.</td>
</tr>
</tbody>
</table>
See Also

ICloneable<T> Interface
Common Namespace
Optimization Framework

ICloneable(T) Methods

ICloneable(T) Interface See Also Send Feedback
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ℹ️ Clone()</td>
<td>Creates a new object that is a copy of the current instance. (Inherited from <code>ICloneable</code>.)</td>
</tr>
<tr>
<td>ℹ️ Clone()</td>
<td>Creates a new object of type $T$ that is a copy of the current instance.</td>
</tr>
</tbody>
</table>
See Also

ICloneable(T) Interface
Common Namespace
ICloneable\(T\).Clone Method
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clone()</td>
<td>Creates a new object that is a copy of the current instance. (Inherited from ICloneable.)</td>
</tr>
<tr>
<td>Clone()</td>
<td>Creates a new object of type $T$ that is a copy of the current instance.</td>
</tr>
</tbody>
</table>
See Also

ICloneable(T) Interface
ICloneable(T) Members
Common Namespace
ICloneable(T).Clone Method

Creates a new object of type $T$ that is a copy of the current instance.

**Namespace:** Common

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>T Clone()</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Function Clone As T</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>T Clone()</code></td>
</tr>
</tbody>
</table>

## Return Value

A new object of type $T$ that is a copy of this instance.
Remarks

Clone can be implemented either as a deep copy or a shallow copy. In a deep copy, all objects are duplicated; whereas, in a shallow copy, only the top-level objects are duplicated and the lower levels contain references.
See Also

ICloneable(T) Interface
Clone Overload
Common Namespace
The core classes you need (Model, Variable, Constraint, etc.) to start modelling
<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConflictingSet</td>
<td></td>
</tr>
<tr>
<td>ConstantExpression</td>
<td>A ConstantExpression only holds a double value but can be used as part of an expression</td>
</tr>
<tr>
<td>Constraint</td>
<td>Represents an constraint</td>
</tr>
<tr>
<td>Expression</td>
<td>An expression essentially is a tree which stores operators on subtrees which can either be Expressions or Terms themselves</td>
</tr>
<tr>
<td>GenericVariableCollectionBase</td>
<td>The base class vor all generic VariableCollection classes</td>
</tr>
<tr>
<td>Model</td>
<td>Represents a mathematical model</td>
</tr>
<tr>
<td>Objective</td>
<td>Represents an objective</td>
</tr>
<tr>
<td>ParameterCollection(T)</td>
<td>A parameterCollection allows you to define parameters on Sets so that you can access parameter values with indices like in other modelling languages</td>
</tr>
<tr>
<td>Solution</td>
<td>Represents a solution of an Model</td>
</tr>
<tr>
<td>SolverConfiguration</td>
<td>Defines common parameters for ISolver instances.</td>
</tr>
<tr>
<td>SumExpressionBuilder</td>
<td>This is a helper class to support efficiently building Expressions from a collection of Terms</td>
</tr>
<tr>
<td>Term</td>
<td>A term represents a variable and a coefficient which belongs to this variable (e.g. 2*x)</td>
</tr>
<tr>
<td>Variable</td>
<td>Represents a variable in an IModel.</td>
</tr>
<tr>
<td>VariableCollection</td>
<td>This is essentially a helper class that allows you to model your Variables more intuitively</td>
</tr>
</tbody>
</table>

Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConflictingSet</td>
<td></td>
</tr>
<tr>
<td>ConstantExpression</td>
<td>A ConstantExpression only holds a double value but can be used as part of an expression</td>
</tr>
<tr>
<td>Constraint</td>
<td>Represents an constraint</td>
</tr>
<tr>
<td>Expression</td>
<td>An expression essentially is a tree which stores operators on subtrees which can either be Expressions or Terms themselves</td>
</tr>
<tr>
<td>GenericVariableCollectionBase</td>
<td>The base class vor all generic VariableCollection classes</td>
</tr>
<tr>
<td>Model</td>
<td>Represents a mathematical model</td>
</tr>
<tr>
<td>Objective</td>
<td>Represents an objective</td>
</tr>
<tr>
<td>ParameterCollection(T)</td>
<td>A parameterCollection allows you to define parameters on Sets so that you can access parameter values with indices like in other modelling languages</td>
</tr>
<tr>
<td>Solution</td>
<td>Represents a solution of an Model</td>
</tr>
<tr>
<td>SolverConfiguration</td>
<td>Defines common parameters for ISolver instances.</td>
</tr>
<tr>
<td>SumExpressionBuilder</td>
<td>This is a helper class to support efficiently building Expressions from a collection of Terms</td>
</tr>
<tr>
<td>Term</td>
<td>A term represents a variable and a coefficient which belongs to this variable (e.g. 2*x)</td>
</tr>
<tr>
<td>Variable</td>
<td>Represents a variable in an IModel.</td>
</tr>
<tr>
<td>VariableCollection</td>
<td>This is essentially a helper class that allows you to model your Variables more intuitively</td>
</tr>
<tr>
<td><code>VariableCollection(T)</code></td>
<td>This is a strongly typed helper class that allows you to model your Variables more intuitively</td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>VariableCollection(T, T2)</code></td>
<td>This is a strongly typed helper class that allows you to model your Variables more intuitively</td>
</tr>
<tr>
<td><code>VariableCollection(T, T2, T3)</code></td>
<td>This is a strongly typed helper class that allows you to model your Variables more intuitively</td>
</tr>
<tr>
<td><code>VariableCollection(T, T2, T3, T4)</code></td>
<td>This is a strongly typed helper class that allows you to model your Variables more intuitively</td>
</tr>
<tr>
<td><code>VariableCollection(T, T2, T3, T4, T5)</code></td>
<td>This is a strongly typed helper class that allows you to model your Variables more intuitively</td>
</tr>
<tr>
<td><code>VariableCollection(T, T2, T3, T4, T5, T6)</code></td>
<td>This is a strongly typed helper class that allows you to model your Variables more intuitively</td>
</tr>
<tr>
<td><code>VariableCollection(T, T2, T3, T4, T5, T6, T7)</code></td>
<td>This is a strongly typed helper class that allows you to model your Variables more intuitively</td>
</tr>
<tr>
<td><code>VariableCollection(T, T2, T3, T4, T5, T6, T7, T8)</code></td>
<td>This is a strongly typed helper class that allows you to model your Variables more intuitively</td>
</tr>
</tbody>
</table>
### Delegates

<table>
<thead>
<tr>
<th>Delegate</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>VariableCollection, BoundGenerator</code></td>
<td>A delegate which takes an array of objects (making up the index) and returns a bound to be used for a variable</td>
</tr>
<tr>
<td><code>VariableCollection, UniqueNameGenerator</code></td>
<td>A delegate which takes an array of objects (making up the index) and returns a <code>StringBuilder</code> which holds a unique id for this index</td>
</tr>
</tbody>
</table>
## Enumerations

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FileType</td>
<td>The filetype of a model file</td>
</tr>
<tr>
<td>ModelBehavior</td>
<td>Describes the behavior of a model when adding constraints</td>
</tr>
<tr>
<td>ObjectiveSense</td>
<td>Sense of an <img src="https://example.com" alt="IObjective" />.</td>
</tr>
<tr>
<td>SOSType</td>
<td>The SOS type</td>
</tr>
<tr>
<td>VariableType</td>
<td>Type of a <img src="https://example.com" alt="IVariable" />.</td>
</tr>
</tbody>
</table>
[Missing <summary> documentation for "T:Optimization.ConflictingSet"]

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public class ConflictingSet</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Class ConflictingSet</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public ref class ConflictingSet</code></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
Optimization.ConflictingSet
See Also

ConflictingSet Members
Optimization Namespace
The `ConflictingSet` type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConflictingSet</td>
<td>Initializes a new instance of the ConflictingSet class.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals</td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
<td></td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
<td></td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
<td></td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
<td></td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
<td></td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
<td></td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConstraintsLB</td>
<td>Gets the conflicting constraints where the lower bound is in conflict.</td>
</tr>
<tr>
<td>ConstraintsUB</td>
<td>Gets the conflicting constraints where the upper bound is in conflict.</td>
</tr>
<tr>
<td>SOS</td>
<td>Gets the conflicting SOS sets.</td>
</tr>
<tr>
<td>VariablesLB</td>
<td>Gets the variables for which the lower bounds are conflicting.</td>
</tr>
<tr>
<td>VariablesUB</td>
<td>Gets the variables for which the upper bounds are conflicting.</td>
</tr>
</tbody>
</table>
See Also

ConflictingSet Class
Optimization Namespace
Initializes a new instance of the `ConflictingSet` class.

**Namespace:** [Optimization](https://example.com)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public ConflictingSet(
    IEnumerable<IVariable> variablesUb,
    IEnumerable<IVariable> variablesLb,
    IEnumerable<Constraint> constraintslb,
    IEnumerable<Constraint> constraintsub,
    IEnumerable<IDictionary<IVariable, double>> sos)
```

#### Visual Basic

```vbnet
Public Sub New (_
    variablesUb As IEnumerable(Of IVariable), _
    variablesLb As IEnumerable(Of IVariable), _
    constraintslb As IEnumerable(Of Constraint),
    constraintsub As IEnumerable(Of Constraint),
    sos As IEnumerable(Of IDictionary(Of IVariable, Double))
)
```

#### Visual C++

```cpp
public:
ConflictingSet(
    IEnumerable<IVariable>^ variablesUb,
    IEnumerable<IVariable>^ variablesLb,
    IEnumerable<Constraint>^ constraintslb,
    IEnumerable<Constraint>^ constraintsub,
    IEnumerable<IDictionary<IVariable^, double>>^ sos)
```

### Parameters
variablesUb
Type: System.Collections.Generic.IEnumerable<IVariable>
The variables for which the upper bounds are conflicting.

variablesLb
Type: System.Collections.Generic.IEnumerable<IVariable>
The variables for which the lower bounds are conflicting.

constraintsLb
Type: System.Collections.Generic.IEnumerable<Constraint>

constraintsSub
Type: System.Collections.Generic.IEnumerable<Constraint>

sos
Type: System.Collections.Generic.IEnumerable<IDictionary<IVariable, Double>)
The conflicting sos stes.
See Also

ConflictingSet Class
Optimization Namespace
The **ConflictingSet** type exposes the following members.
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Equals**    | Determines whether the specified [Object](#) is equal to the current [Object](#).  
(Inherited from [Object](#).)                                           |
| **Finalize**  | Allows an [Object](#) to attempt to free resources and perform other cleanup  
operations before the [Object](#) is reclaimed by garbage collection.  
(Inherited from [Object](#).)                                           |
| **GetHashCode** | Serves as a hash function for a particular type.  
(Inherited from [Object](#).)                                           |
| **GetType**   | Gets the [Type](#) of the current instance.  
(Inherited from [Object](#).)                                           |
| **MemberwiseClone** | Creates a shallow copy of the current [Object](#).  
(Inherited from [Object](#).)                                           |
| **ToString**  | Returns a [String](#) that represents the current [Object](#).  
(Inherited from [Object](#).)                                           |
See Also

ConflictingSet Class
Optimization Namespace
The **ConflictingSet** type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConstraintsLB</td>
<td>Gets the conflicting constraints where the lower bound is in conflict.</td>
</tr>
<tr>
<td>ConstraintsUB</td>
<td>Gets the conflicting constraints where the upper bound is in conflict.</td>
</tr>
<tr>
<td>SOS</td>
<td>Gets the conflicting SOS sets.</td>
</tr>
<tr>
<td>VariablesLB</td>
<td>Gets the variables for which the lower bounds are conflicting.</td>
</tr>
<tr>
<td>VariablesUB</td>
<td>Gets the variables for which the upper bounds are conflicting.</td>
</tr>
</tbody>
</table>
See Also

ConflictingSet Class
Optimization Namespace
Gets the conflicting constraints where the lower bound is in conflict.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#**   | ```
public IEnumerable<Constraint> ConstraintsLB { get; } |
| **Visual Basic** | ```Public Property ConstraintsLB As IEnumerable(Of Constraint)
Get
Private Set |
| **Visual C++** | ```
public:
pROPERTY IEnumerable<Constraint>^ ConstraintsLB { 
IEnumerable<Constraint>^ get (); 
private: void set (IEnumerable<Constraint>^) 
} ```
See Also

ConflictingSet Class
Optimization Namespace
Gets the conflicting constraints where the upper bound is in conflict.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td><code>public IEnumerable&lt;Constraint&gt; ConstraintsUB { get; }</code></td>
</tr>
<tr>
<td>Visual Basic</td>
<td><code>Public Property ConstraintsUB As IEnumerable(Of Constraint)</code></td>
</tr>
</tbody>
</table>
| Visual C++ | `public:
  property IEnumerable<Constraint>^ ConstraintsUB { IEnumerable<Constraint>^ get ();
  private: void set (IEnumerable<Constraint>^)` |
See Also

ConflictingSet Class
Optimization Namespace
Optimization Framework

ConflictingSet.SOS Property

ConflictingSet Class  See Also  Send Feedback

Gets the conflicting SOS sets.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>public <code>IEnumerable&lt;IDictionary&lt;IVariable, double&gt;&gt;</code> SOS</td>
</tr>
<tr>
<td>Visual Basic</td>
<td><code>Public Property SOS As IEnumerable(Of IDictionary(Of IVariable, double))</code></td>
</tr>
<tr>
<td>Visual C++</td>
<td><code>public: property IEnumerable&lt;IDictionary&lt;IVariable^, double^, double^, double^&gt;^ IEnumerable&lt;IDictionary&lt;IVariable^, double^, double^, double^&gt;^: private: void set (IEnumerable&lt;IDictionary&lt;IVariable^, double^, double^, double^&gt;^)</code></td>
</tr>
</tbody>
</table>
See Also

ConflictingSet Class
Optimization Namespace
Optimization Framework

ConflictingSet.VariablesLB Property

Gets the variables for which the lower bounds are conflicting.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>public IEnumerable&lt;IVariable&gt; VariablesLB { get; private; }</td>
</tr>
<tr>
<td>Visual Basic</td>
<td>Public Property VariablesLB As IEnumerable(Of IVariable) Get Private Set</td>
</tr>
<tr>
<td>Visual C++</td>
<td>public: property IEnumerable&lt;IVariable^&gt;^ VariablesLB { IEnumerable&lt;IVariable^&gt;^ get (); private: void set (IEnumerable&lt;IVariable^&gt;^); }</td>
</tr>
</tbody>
</table>
See Also

ConflictingSet Class
Optimization Namespace
ConflictingSet.VariablesUB Property

Gets the variables for which the upper bounds are conflicting

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
</tr>
</thead>
<tbody>
<tr>
<td>public <code>IEnumerable&lt;IVariable&gt;</code> VariablesUB { get; private; }</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual Basic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Property VariablesUB As <code>IEnumerable(Of IVariable)</code></td>
</tr>
<tr>
<td>Get</td>
</tr>
<tr>
<td>Private Set</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>public: property <code>IEnumerable&lt;IVariable^&gt;</code> VariablesUB {</td>
</tr>
<tr>
<td><code>IEnumerable&lt;IVariable^&gt;</code> get ();</td>
</tr>
<tr>
<td>private: void set (<code>IEnumerable&lt;IVariable^&gt;</code> value)</td>
</tr>
<tr>
<td>}</td>
</tr>
</tbody>
</table>
See Also

ConflictingSet Class
Optimization Namespace
A `ConstantExpression` only holds a double value but can be used as part of an expression.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public class ConstantExpression : Expression
```

**Visual Basic**

```vbnet
Public Class ConstantExpression _
    Inherits Expression
```

**Visual C++**

```cpp
public ref class ConstantExpression : public Expression
```
Inheritance Hierarchy

System.Object
Optimization.Expression
Optimization.ConstantExpression
See Also

ConstantExpression Members
Optimization Namespace
The **ConstantExpression** type exposes the following members.
### Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ConstantExpression</code></td>
<td>Initializes a new instance of the <code>ConstantExpression</code> class.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>Evaluate</strong></td>
<td>Evaluates the expression using the specified variable values. (Inherited from <code>Expression</code>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <code>String</code> that represents this instance. (Override <code>Object.ToString</code>.)</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>Addition(Double, ConstantExpression)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(ConstantExpression, Double)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Implicit(Double to ConstantExpression)</td>
<td>Performs an implicit conversion from Double to ConstantExpression.</td>
</tr>
<tr>
<td>Implicit(Int32 to ConstantExpression)</td>
<td>Performs an implicit conversion from Int32 to ConstantExpression.</td>
</tr>
<tr>
<td>Multiply(Double, ConstantExpression)</td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td>Multiply(ConstantExpression, Double)</td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td>Subtraction(Double, ConstantExpression)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(ConstantExpression, Double)</td>
<td>Implements the operator +.</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Constant</strong></td>
<td>Gets the constant. (Overrides Expression.Constant.)</td>
</tr>
<tr>
<td><strong>ExpressionLowerEstimate</strong></td>
<td>Gives a lower estimate on the value of an Expression. ToDo: Very experimental. Should be able to estimate linear combinations of variables with finite bounds. In the default case -infinity is returned! (Inherited from Expression.)</td>
</tr>
<tr>
<td><strong>ExpressionUpperEstimate</strong></td>
<td>Gives an upper estimate in the value of an Expression. ToDo: Very experimental. Should be able to estimate linear combinations of variables with finite bounds. In the default case -infinity is returned! (Inherited from Expression.)</td>
</tr>
<tr>
<td><strong>isLinear</strong></td>
<td>Gets or sets a value indicating whether this instance is linear. (Overides Expression.isLinear.)</td>
</tr>
<tr>
<td><strong>Terms</strong></td>
<td>Gets the terms. (Overides Expression.Terms.)</td>
</tr>
<tr>
<td><strong>Variables</strong></td>
<td>Gets the variables. (Overides Expression.Variables.)</td>
</tr>
</tbody>
</table>
See Also

ConstantExpression Class
Optimization Namespace
Optimization Framework

**ConstantExpression Constructor**

Initializes a new instance of the `ConstantExpression` class.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#**   | ```
    public ConstantExpression(
        double value
    )
``` |
| **Visual Basic** | ```
    Public Sub New ( _
        value As Double _
    )
``` |
| **Visual C++** | ```
    public:
    ConstantExpression(
        double value
    )
``` |

### Parameters

*value*

Type: [System.Double](https://docs.microsoft.com/en-us/dotnet/api/system.double)

The value.
See Also

ConstantExpression Class
Optimization Namespace
The `ConstantExpression` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Evaluate</strong></td>
<td>Evaluates the expression using the specified variable values. (Inherited from <a href="#">Expression</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents this instance. (Overides <a href="#">Object.ToString()</a>.)</td>
</tr>
</tbody>
</table>
See Also

ConstantExpression Class
Optimization Namespace
Returns a `String` that represents this instance.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

<table>
<thead>
<tr>
<th></th>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><code>public override string ToString()</code></td>
<td><code>Public Overrides Function ToString As String</code></td>
<td><code>public: virtual String^ ToString() override</code></td>
</tr>
</tbody>
</table>

**Return Value**

A `String` that represents this instance.
## Contracts

**Ensures**

`Contract.Result<string>() != null`

*Inherited from:* [Object](#)

[Learn more about contracts](#)
See Also

ConstantExpression Class
Optimization Namespace
Optimization Framework

ConstantExpression Operators and Type Conversions

ConstantExpression Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addition(Double, ConstantExpression)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(ConstantExpression, Double)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Implicit(Double to ConstantExpression)</td>
<td>Performs an implicit conversion from Double to ConstantExpression.</td>
</tr>
<tr>
<td>Implicit(Int32 to ConstantExpression)</td>
<td>Performs an implicit conversion from Int32 to ConstantExpression.</td>
</tr>
<tr>
<td>Multiply(Double, ConstantExpression)</td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td>Multiply(ConstantExpression, Double)</td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td>Subtraction(Double, ConstantExpression)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(ConstantExpression, Double)</td>
<td>Implements the operator +.</td>
</tr>
</tbody>
</table>
See Also

ConstantExpression Class
Optimization Namespace
ConstantExpression.Addition Operator
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addition(Double, ConstantExpression)</td>
<td>Implements the operator <code>+</code>.</td>
</tr>
<tr>
<td>Addition(ConstantExpression, Double)</td>
<td>Implements the operator <code>+</code>.</td>
</tr>
</tbody>
</table>
See Also

ConstantExpression Class
ConstantExpression Members
Optimization Namespace
Implements the operator `+`.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#** | ```
public static ConstantExpression operator +(  
double constant,
ConstantExpression expression
)
``` |
| **Visual Basic** | ```
Public Shared Operator + (   
    constant As Double,   
    expression As ConstantExpression  
) As ConstantExpression
``` |
| **Visual C++** | ```
public:  
static ConstantExpression^ operator +(  
double constant,
ConstantExpression^ expression
)
``` |

### Parameters

- **constant**
  - Type: `System.Double`
  - The constant.

- **expression**
  - Type: `Optimization.ConstantExpression`
  - The expression.

### Return Value
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!object.Equals(expression, null)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;ConstantExpression&gt;() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

ConstantExpression Class
Addition Overload
Optimization Namespace
ConstantExpression.Addition Operator (ConstantExpression, Double)

Implements the operator +.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

<table>
<thead>
<tr>
<th>C#</th>
</tr>
</thead>
<tbody>
<tr>
<td>public static ConstantExpression operator +(</td>
</tr>
<tr>
<td>ConstantExpression expression,</td>
</tr>
<tr>
<td>double constant</td>
</tr>
<tr>
<td>)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual Basic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Shared Operator + ( _ expression As ConstantExpression, _</td>
</tr>
<tr>
<td>constant As Double _ ) As ConstantExpression</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>public:</td>
</tr>
<tr>
<td>static ConstantExpression^ operator +(</td>
</tr>
<tr>
<td>ConstantExpression^ expression,</td>
</tr>
<tr>
<td>double constant</td>
</tr>
<tr>
<td>)</td>
</tr>
</tbody>
</table>

**Parameters**

*expression*

Type: Optimization,ConstantExpression

The expression.

*constant*

Type: System,Double

The constant.

**Return Value**
The result of the operator.
<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!object.Equals(expression, null)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;ConstantExpression&gt;() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

ConstantExpression Class
Addition Overload
Optimization Namespace
Optimization Framework

**ConstantExpression Implicit Conversion Operators**

[ConstantExpression Class](#)  [See Also](#)  [Send Feedback](#)
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implicit(Double to ConstantExpression)</td>
<td>Performs an implicit conversion from Double to ConstantExpression.</td>
</tr>
<tr>
<td>Implicit(Int32 to ConstantExpression)</td>
<td>Performs an implicit conversion from Int32 to ConstantExpression.</td>
</tr>
</tbody>
</table>
See Also

ConstantExpression Class
ConstantExpression Members
Optimization Namespace
Perform an implicit conversion from Double to ConstantExpression.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
</table>
| public static implicit operator **ConstantExpression** (  
  **double** value  
) | Public Shared Widening Operator **CType** ( _  
  value As **Double** _  
) As **ConstantExpression** | static implicit operator **ConstantExpression**^ (  
  **double** value  
) |

### Parameters

**value**  
Type: **System.Double**  
The value.

### Return Value

The result of the conversion.
See Also

ConstantExpression Class
Implicit Overload
Optimization Namespace
Performs an implicit conversion from \texttt{Int32} to \texttt{ConstantExpression}.

\textbf{Namespace:} \texttt{Optimization}
\textbf{Assembly:} Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

**C#**

```csharp
public static implicit operator ConstantExpression (int value)
```

**Visual Basic**

```vbnet
Public Shared Widening Operator CType (_
    value As Integer _
) As ConstantExpression
```

**Visual C++**

```cpp
static implicit operator ConstantExpression^ (int value)
```

**Parameters**

`value`
Type: `System.Int32`
The value.

**Return Value**
The result of the conversion.
See Also

ConstantExpression Class
Implicit Overload
Optimization Namespace
ConstantExpression.Multiply Operator

ConstantExpression Class  See Also  Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Multiply(Double, ConstantExpression)</code></td>
<td>Implements the operator <code>*</code>.</td>
</tr>
<tr>
<td><code>Multiply(ConstantExpression, Double)</code></td>
<td>Implements the operator <code>*</code>.</td>
</tr>
</tbody>
</table>
See Also

ConstantExpression Class
ConstantExpression Members
Optimization Namespace
Optimization Framework

**ConstantExpression.Multiply Operator (Double, ConstantExpression)**

See Also Send Feedback

Implements the operator `*`.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
C#

```csharp
public static ConstantExpression operator *(double factor, ConstantExpression expression)
```

Visual Basic

```vbnet
Public Shared Operator *(factor As Double, expression As ConstantExpression) As ConstantExpression
```

Visual C++

```cpp
public:
static ConstantExpression^ operator *(double factor, ConstantExpression^ expression)
```

**Parameters**

*factor*
Type: `System.Double`
The factor.

*expression*
Type: `Optimization.ConstantExpression`
The constant expression.

**Return Value**
A new ConstantExpression.
Contracts

Requires
!object.Equals(expression, null)

Ensures
(Object)Contract.Result<ConstantExpression>() != null

Learn more about contracts
See Also

ConstantExpression Class
Multiply Overload
Optimization Namespace
Implements the operator *.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>public static <code>ConstantExpression</code> operator *(</td>
</tr>
<tr>
<td></td>
<td><code>ConstantExpression</code> expression,</td>
</tr>
<tr>
<td></td>
<td><code>double</code> factor</td>
</tr>
<tr>
<td></td>
<td>)</td>
</tr>
<tr>
<td>Visual Basic</td>
<td>Public Shared Operator * (</td>
</tr>
<tr>
<td></td>
<td><code>_expression As ConstantExpression, _factor As Double _</code></td>
</tr>
<tr>
<td></td>
<td>) As ConstantExpression</td>
</tr>
<tr>
<td>Visual C++</td>
<td>public:</td>
</tr>
<tr>
<td></td>
<td>static <code>ConstantExpression</code> operator *(</td>
</tr>
<tr>
<td></td>
<td><code>ConstantExpression</code> expression,</td>
</tr>
<tr>
<td></td>
<td><code>double</code> factor</td>
</tr>
<tr>
<td></td>
<td>)</td>
</tr>
</tbody>
</table>

### Parameters

- **expression**
  - Type: `Optimization.ConstantExpression`
  - The expression.

- **factor**
  - Type: `System.Double`
  - The factor.

### Return Value
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>![object.Equals(expression, null)](expression, null)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;ConstantExpression&gt;() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

ConstantExpression Class
Multiply Overload
Optimization Namespace
Optimization Framework

ConstantExpression.Subtraction Operator

ConstantExpression Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subtraction(Double, ConstantExpression)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(ConstantExpression, Double)</td>
<td>Implements the operator +.</td>
</tr>
</tbody>
</table>
See Also

ConstantExpression Class
ConstantExpression Members
Optimization Namespace
Optimization Framework

ConstantExpression.Subtraction Operator (Double, ConstantExpression)

ConstantsExpression Class  See Also  Send Feedback

Implements the operator -.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static ConstantExpression operator -(double constant, ConstantExpression expression)
```

**Visual Basic**

```vbnet
Public Shared Operator -( _
    constant As Double, _
    expression As ConstantExpression _
) As ConstantExpression
```

**Visual C++**

```cpp
public:
static ConstantExpression^ operator -(double constant, ConstantExpression^ expression)
```

#### Parameters

- **constant**
  - Type: `System.Double`
  - The constant.

- **expression**
  - Type: `Optimization.ConstantExpression`
  - The expression.

#### Return Value
The result of the operator.
**Contracts**

<table>
<thead>
<tr>
<th>Requires</th>
<th>!object.Equals(expression, null)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ensures</td>
<td>(Object)Contract.Result&lt;ConstantExpression&gt;() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

ConstantExpression Class
Subtraction Overload
Optimization Namespace
Optimization Framework

ConstantExpression.Subtraction Operator (ConstantExpression, Double)

ConstantExpression Class  See Also  Send Feedback

Implements the operator +.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public static ConstantExpression operator -(ConstantExpression expression, double constant)
```

#### Visual Basic

```vbnet
Public Shared Operator -( _
    expression As ConstantExpression, _
    constant As Double _
) As ConstantExpression
```

#### Visual C++

```cpp
public:
static ConstantExpression ^ operator -(ConstantExpression ^ expression, double constant)
```

### Parameters

**expression**
- Type: `Optimization.ConstantExpression`
- The expression.

**constant**
- Type: `System.Double`
- The constant.

### Return Value
The result of the operator.
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!object.Equals(expression, null)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;ConstantExpression&gt;() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

ConstantExpression Class
Subtraction Overload
Optimization Namespace
The **ConstantExpression** type exposes the following members.
# Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Constant</strong></td>
<td>Gets the constant. (Overrides Expression_CONSTANT.)</td>
</tr>
<tr>
<td><strong>ExpressionLowerEstimate</strong></td>
<td>Gives a lower estimate on the value of an Expression. ToDo: Very experimental. Should be able to estimate linear combinations of variables with finite bounds. In the default case -infinity is returned! (Inherited from Expression.)</td>
</tr>
<tr>
<td><strong>ExpressionUpperEstimate</strong></td>
<td>Gives an upper estimate in the value of an Expression. ToDo: Very experimental. Should be able to estimate linear combinations of variables with finite bounds. In the default case -infinity is returned! (Inherited from Expression.)</td>
</tr>
<tr>
<td><strong>isLinear</strong></td>
<td>Gets or sets a value indicating whether this instance is linear. (Overrides Expression.isLinear.)</td>
</tr>
<tr>
<td><strong>Terms</strong></td>
<td>Gets the terms. (Overides Expression.Terms.)</td>
</tr>
<tr>
<td><strong>Variables</strong></td>
<td>Gets the variables. (Overides Expression.Variables.)</td>
</tr>
</tbody>
</table>
See Also

ConstantExpression Class
Optimization Namespace
Gets the constant.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
</tr>
</thead>
<tbody>
<tr>
<td>public override double Constant { get; }</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual Basic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Overrides ReadOnly Property Constant As Double</td>
</tr>
<tr>
<td>Get</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>public:</td>
</tr>
<tr>
<td>virtual property double Constant {</td>
</tr>
<tr>
<td>double get () override;</td>
</tr>
<tr>
<td>}</td>
</tr>
</tbody>
</table>
See Also

ConstantExpression Class
Optimization Namespace
Gets or sets a value indicating whether this instance is linear.

**Namespace:** [Optimization](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>public override bool isLinear { get; }</td>
</tr>
<tr>
<td>Visual Basic</td>
<td>Public Overrides ReadOnly Property isLinear As Boolean Get</td>
</tr>
<tr>
<td>Visual C++</td>
<td>public: virtual property bool isLinear { bool get () override; }</td>
</tr>
</tbody>
</table>

**Field Value**

Always `true` because a [ConstantExpression](#) is always linear.
See Also

- ConstantExpression Class
- Optimization Namespace
Optimization Framework

ConstantExpression.Terms Property

Gets the terms.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

**C#**

```csharp
public override IEnumerable<Term> Terms { get; }
```

**Visual Basic**

```vbnet
Public Overrides ReadOnly Property Terms As IEnumerable(Term)
Get
```

**Visual C++**

```cpp
public:
virtual property IEnumerable<Term^>^ Terms {
IEnumerable<Term^>^ get () override;
}
```

**Field Value**

Always **null** because a [ConstantExpression](#) has no Terms.
See Also

ConstantExpression Class
Optimization Namespace
Gets the variables.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public override IEnumerable<IVariable> Variables { get }
```

Visual Basic

```vbnet
Public Overrides ReadOnly Property Variables As IEnumerable.Get
```

Visual C++

```cpp
public:
virtual property IEnumerable<IVariable>^ Variables {
IEnumerable<IVariable>^ get () override;
}
```

Field Value

Always **null** because a ConstantExpression has no Variables.
See Also

ConstantExpression Class
Optimization Namespace
Represents an constraint

**Namespace:** [Optimization](https://example.com)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>[SerializableAttribute]</code> public class Constraint</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>`&lt;SerializableAttribute&gt; _ Public Class Constraint</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>[SerializableAttribute]</code> public ref class Constraint</td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
Optimization.Constraint
See Also

Constraint Members
Optimization Namespace
The **Constraint** type exposes the following members.
### Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td>Initializes a new instance of the Constraint class.</td>
</tr>
</tbody>
</table>
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Equals(Object)**          | Determines whether the specified **Object** is equal to the current **Object**.  
(Inherited from **Object**. ) |
| **Equals(Expression, Expression)** | Compares the specified expressions. |
| **Equals(Expression, Double)** | Compares the expression to a constant. |
| **Finalize**                | Allows an **Object** to attempt to free resources and perform other cleanup operations before the **Object** is reclaimed by garbage collection.  
(Inherited from **Object**. ) |
| **GetHashCode**            | Serves as a hash function for a particular type.  
(Inherited from **Object**. ) |
| **GetType**                | Gets the **Type** of the current instance.  
(Inherited from **Object**. ) |
<p>| <strong>GreaterThanOrEqual(Expression, Expression)</strong> | Compares two expressions whether the first expression is greater or equal the second expression. |
| <strong>GreaterThanOrEqual(Expression, Double)</strong> | Compares the expression to a constant whether its greater or equal. |
| <strong>LessThanOrEqual(Expression, Expression)</strong> | Compares two expressions whether the first expression is lower or equal the second expression. |
| <strong>LessThanOrEqual(Expression, Double)</strong> | Compares the expression to a constant whether its less or |</p>
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a String that represents this instance. (Overrides Object.ToString().)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Expression</strong></td>
<td>Expression of this constraint.</td>
</tr>
<tr>
<td><strong>LowerBound</strong></td>
<td>Lower bound (left hand side) of this constraint.</td>
</tr>
<tr>
<td><strong>Name</strong></td>
<td>Unique name of this constraint.</td>
</tr>
<tr>
<td><strong>StochasticStage</strong></td>
<td>StochasticStage of this constraint.</td>
</tr>
<tr>
<td><strong>UpperBound</strong></td>
<td>Upper bound (right hand side) of this constraint.</td>
</tr>
</tbody>
</table>
See Also

Constraint Class
Optimization Namespace
Initializes a new instance of the `Constraint` class.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public Constraint(
    Expression expression,
    string name,
    double lowerBound,
    double upperBound,
    int stochasticStage
)
```

#### Visual Basic

```vbnet
Public Sub New ( _
    expression As Expression, _
    name As String, _
    lowerBound As Double, _
    upperBound As Double, _
    stochasticStage As Integer _
)
```

#### Visual C++

```cpp
public:
Constraint(
    Expression^ expression,
    String^ name,
    double lowerBound,
    double upperBound,
    int stochasticStage
)
```

## Parameters
expression
Type: Optimization.Expression
The expression.

name
Type: System.String
The name.

lowerBound
Type: System.Double
The lower bound.

upperBound
Type: System.Double
The upper bound.

stochasticStage
Type: System.Int32
The stochastic stage of this constraint
See Also

Constraint Class
Optimization Namespace
The **Constraint** type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals(Object)</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Equals(Expression, Expression)</strong></td>
<td>Compares the specified expressions.</td>
</tr>
<tr>
<td><strong>Equals(Expression, Double)</strong></td>
<td>Compares the expression to a constant.</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GreaterThanOrEqual(Expression, Expression)</strong></td>
<td>Compares two expressions whether the first expression is greater or equal the second expression.</td>
</tr>
<tr>
<td><strong>GreaterThanOrEqual(Expression, Double)</strong></td>
<td>Compares the expression to a constant whether its greater or equal.</td>
</tr>
<tr>
<td><strong>LessThanOrEqual(Expression, Expression)</strong></td>
<td>Compares two expressions whether the first expression is lower or equal the second expression.</td>
</tr>
</tbody>
</table>
| **LessThanOrEqual(Expression, Double)**    | Compares the expression to a constant whether its less or
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>MemberwiseClone</code></td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><code>ToString</code></td>
<td>Returns a <code>String</code> that represents this instance. (Overrides <code>Object.ToString()</code>.)</td>
</tr>
</tbody>
</table>
See Also

Constraint Class
Optimization Namespace
Optimization Framework

**Constraint.Equals Method**

[Constraint Class][See Also][Send Feedback]
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>🌃 <strong>Equals(Object)</strong></td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>. <em>(Inherited from <strong>Object</strong>.)</em></td>
</tr>
<tr>
<td>🌃 <strong>Equals(Expression, Expression)</strong></td>
<td>Compares the specified expressions.</td>
</tr>
<tr>
<td>🌃 <strong>Equals(Expression, Double)</strong></td>
<td>Compares the expression to a constant.</td>
</tr>
</tbody>
</table>
See Also

Constraint Class
Constraint Members
Optimization Namespace
Constraint.Equals Method (Expression, Expression)

Compares the specified expressions.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public static Constraint Equals(
    Expression expression1,
    Expression expression2
)
```

#### Visual Basic

```vbnet
Public Shared Function Equals (
    expression1 As Expression,
    expression2 As Expression
) As Constraint
```

#### Visual C++

```cpp
public:
    static Constraint^ Equals(
        Expression^ expression1,
        Expression^ expression2
    )
```

### Parameters

- **expression1**
  - Type: `Optimization.Expression`
  - The first expression.

- **expression2**
  - Type: `Optimization.Expression`
  - The second expression.

### Return Value
A Constraint with lower and upper bound of zero
## Contracts

### Requires

<table>
<thead>
<tr>
<th>Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>(Object)expression1 != null</code></td>
<td></td>
</tr>
<tr>
<td><code>(Object)expression2 != null</code></td>
<td></td>
</tr>
</tbody>
</table>

### Ensures

<table>
<thead>
<tr>
<th>Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Contract.Result&lt;Constraint&gt;() != null</code></td>
<td></td>
</tr>
<tr>
<td><code>Contract.Result&lt;Constraint&gt;().LowerBound == 0</code></td>
<td><em>Description:</em> The lower bound of the constraint is 0</td>
</tr>
<tr>
<td><code>Contract.Result&lt;Constraint&gt;().UpperBound == 0</code></td>
<td><em>Description:</em> The upper bound of the constraint is 0</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Constraint Class
Equals Overload
Optimization Namespace
Constraint.Equals Method (Expression, Double)

Compares the expression to a constant.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
class Constraint

public static Constraint Equals(
    Expression expression1,
    double constant
)
```

#### Visual Basic

```vbnet
Public Shared Function Equals ( _
    expression1 As Expression, _
    constant As Double _
) As Constraint
```

#### Visual C++

```cpp
public:
static Constraint^ Equals(
    Expression^ expression1,
    double constant
)
```

### Parameters

**expression1**

Type: `Optimization.Expression`

The expression.

**constant**

Type: `System.Double`

The constant.

### Return Value
A Constraint with lower and upper bound of zero
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>(Object)expression1 != null</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Contract.Result&lt;Constraint&gt;() != null</code></td>
</tr>
<tr>
<td><code>Contract.Result&lt;Constraint&gt;().LowerBound == 0</code></td>
</tr>
</tbody>
</table>
  *Description:* The lower bound of the constraint is 0
| `Contract.Result<Constraint>().UpperBound == 0` |
  *Description:* The upper bound of the constraint is 0

[Learn more about contracts](#)
See Also

Constraint Class
Equals Overload
Optimization Namespace
Optimization Framework

**Constraint.GreaterThanOrEqual Method**

[Constraint Class][See Also][Send Feedback]
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GreaterThanOrEqual(Expression, Expression)</td>
<td>Compares two expressions whether the first expression is greater or equal the second expression.</td>
</tr>
<tr>
<td>GreaterThanOrEqual(Expression, Double)</td>
<td>Compares the expression to a constant whether its greater or equal.</td>
</tr>
</tbody>
</table>
See Also

Constraint Class
Constraint Members
Optimization Namespace
Constraint.GreaterThanOrEqual Method (Expression, Expression)

Compares two expressions whether the first expression is greater or equal the second expression.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public static Constraint GreaterThanOrEqual(
    Expression expression1,
    Expression expression2
)
```

Visual Basic

```vbnet
Public Shared Function GreaterThanOrEqual ( _
    expression1 As Expression, _
    expression2 As Expression _
) As Constraint
```

Visual C++

```cpp
public:
static Constraint^ GreaterThanOrEqual(
    Expression^ expression1,
    Expression^ expression2
)
```

Parameters

- **expression1**
  Type: **Optimization.Expression**
  The first expression.

- **expression2**
  Type: **Optimization.Expression**
  The second expression.

Return Value
A Constraint with lower bound of zero
Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)expression1 != null</td>
</tr>
<tr>
<td>(Object)expression2 != null</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contract.Result&lt;Constraint&gt;() != null</td>
</tr>
<tr>
<td>Contract.Result&lt;Constraint&gt;().LowerBound == 0</td>
</tr>
</tbody>
</table>

Description: The lower bound of the constraint is 0

Learn more about contracts
See Also

Constraint Class
GreaterThanOrEqual Overload
Optimization Namespace
Optimization Framework

Constraint.GreaterThanOrEqual Method (Expression, Double)

Compares the expression to a constant whether its greater or equal.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Parameters**

*expression1*
Type: [Optimization.Expression](#)
The expression.

*constant*
Type: [System.Double](#)
The constant.

**Return Value**
A Constraint with lower bound of zero
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)expression1 != null</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contract.Result&lt;Constraint&gt;().LowerBound == 0</td>
</tr>
</tbody>
</table>

*Description: The lower bound of the constraint is 0*

[Learn more about contracts](#)
See Also

Constraint Class
GreaterThanOrEqual Overload
Optimization Namespace
Constraint.LessThanOrEqual Method

Constraint Class  See Also  Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>LessThanOrEqual(Expression, Expression)</code></td>
<td>Compares two expressions whether the first expression is lower or equal the second expression.</td>
</tr>
<tr>
<td><code>LessThanOrEqual(Expression, Double)</code></td>
<td>Compares the expression to a constant whether its less or equal.</td>
</tr>
</tbody>
</table>
See Also

Constraint Class
Constraint Members
Optimization Namespace
Optimization Framework

**Constraint.LessThanOrEqual Method (Expression, Expression)**

*Constraint Class*  *See Also*  *Send Feedback*

Compares two expressions whether the first expression is lower or equal the second expression.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public static Constraint LessThanOrEqual(
    Expression expression1,
    Expression expression2
)
```

#### Visual Basic

```vbnet
Public Shared Function LessThanOrEqual ( _
    expression1 As Expression, _
    expression2 As Expression _
) As Constraint
```

#### Visual C++

```c++
public:
static Constraint^ LessThanOrEqual(
    Expression^ expression1,
    Expression^ expression2
)
```

### Parameters

**expression1**
Type: `Optimization.Expression`
The first expression.

**expression2**
Type: `Optimization.Expression`
The second expression.

### Return Value
A Constraint with upper bound of zero
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>(Object)expression1 != null</code></td>
</tr>
<tr>
<td><code>(Object)expression2 != null</code></td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Constraint Class
LessThanOrEqual Overload
Optimization Namespace
Constraint.LessThanOrEqual Method (Expression, Double)

Compares the expression to a constant whether its less or equal.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th></th>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public static Constraint LessThanOrEqual(</td>
<td>Public Shared Function LessThanOrEqual ( _</td>
<td>public: static Constraint^ LessThanOrEqual(</td>
</tr>
<tr>
<td></td>
<td>Expression expression1,</td>
<td>expression1 As Expression, _</td>
<td>Expression^ expression1,</td>
</tr>
<tr>
<td></td>
<td>double constant</td>
<td>constant As Double _</td>
<td>double constant</td>
</tr>
<tr>
<td></td>
<td>)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Parameters

- **expression1**
  - Type: **Optimization.Expression**
  - The expression.

- **constant**
  - Type: **System.Double**
  - The constant.

### Return Value
A Constraint with upper bound of zero
<table>
<thead>
<tr>
<th><strong>Contracts</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Requires</strong></td>
</tr>
<tr>
<td><code>(Object)expression1 != null</code></td>
</tr>
<tr>
<td><strong>Ensures</strong></td>
</tr>
<tr>
<td><code>Contract.Result&lt;Constraint&gt;() != null</code></td>
</tr>
<tr>
<td><code>Contract.Result&lt;Constraint&gt;().UpperBound == 0</code></td>
</tr>
</tbody>
</table>

**Description:** The upper bound of the constraint is 0

[Learn more about contracts](#)
See Also

Constraint Class
LessThanOrEqual Overload
Optimization Namespace
Constraint.ToString Method

Returns a `String` that represents this instance.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public override string ToString()</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Overrides Function ToString As String</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public: virtual String^ ToString() override</code></td>
</tr>
</tbody>
</table>

**Return Value**

A [String](#) that represents this instance.
<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Contract.Result&lt;string&gt;() != null</code></td>
</tr>
</tbody>
</table>

*Inherited from:* [Object](#)

[Learn more about contracts](#)
See Also

Constraint Class
Optimization Namespace
The `Constraint` type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expression</td>
<td>Expression of this constraint.</td>
</tr>
<tr>
<td>LowerBound</td>
<td>Lower bound (left hand side) of this constraint.</td>
</tr>
<tr>
<td>Name</td>
<td>Unique name of this constraint.</td>
</tr>
<tr>
<td>StochasticStage</td>
<td>StochasticStage this of this constraint</td>
</tr>
<tr>
<td>UpperBound</td>
<td>Upper bound (right hand side) of this constraint.</td>
</tr>
</tbody>
</table>

See Also

Constraint Class
Optimization Namespace
Expression of this constraint.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public Expression Expression { get; set; }
```

#### Visual Basic

```vbnet
Public Property Expression As Expression
    Get
    Set
```

#### Visual C++

```cpp
public:
    property Expression^ Expression { 
        Expression^ get ();
        void set (Expression^ value);
    }
```

### Field Value
See Also

Constraint Class
Optimization Namespace
Lower bound (left hand side) of this constraint.

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public double LowerBound { get; set; }
```

### Visual Basic

```vbnet
Public Property LowerBound As Double
    Get
    Set
```

### Visual C++

```cpp
public:
    property double LowerBound {
        double get ();
        void set (double value);
    }
```

## Field Value
See Also

Constraint Class
Optimization Namespace
Unique name of this constraint.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td><code>public string Name { get; set; }</code></td>
</tr>
</tbody>
</table>
| Visual Basic | `Public Property Name As String
     Get
     Set` |
| Visual C++ | `public:
  property String^ Name {
    String^ get ();
    void set (String^ value);
  }` |

### Field Value
See Also

Constraint Class
Optimization Namespace
StochasticStage this of this constraint

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public int StochasticStage { get; set; }
```

**Visual Basic**

```vbnet
Public Property StochasticStage As Integer
    Get
        Set
    End Property
```

**Visual C++**

```cpp
public:
    property int StochasticStage {
        int get ();
        void set (int value);
    }
```
See Also

Constraint Class
Optimization Namespace
Upper bound (right hand side) of this constraint.

**Namespace**: Optimization

**Assembly**: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public double UpperBound { get; set; }
```

### Visual Basic

```vbnet
Public Property UpperBound As Double
    Get
    Set
```

### Visual C++

```cpp
public:
    property double UpperBound {
        double get ();
        void set (double value);
    }
```

## Field Value
See Also

Constraint Class
Optimization Namespace
An expression essentially is a tree which stores operators on subtrees which can either be Expressions or Terms themselves

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>[SerializableAttribute]</code>&lt;br&gt;<code>public abstract class Expression</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>&lt;SerializableAttribute&gt; _</code>&lt;br&gt;<code>Public MustInherit Class Expression</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>[SerializableAttribute]</code>&lt;br&gt;<code>public ref class Expression abstract</code></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

- System.Object
- Optimization.Expression
- Optimization.ConstantExpression
- Optimization.Term
See Also

Expression Members
Optimization Namespace
The **Expression** type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expression</td>
<td>Initializes a new instance of the Expression class</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is equal to the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Evaluate</strong></td>
<td>Evaluates the expression using the specified variable values.</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.type">Type</a> of the current instance. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Sum(IEnumerable&lt;Double&gt;())</strong></td>
<td>Sums up the specified values.</td>
</tr>
<tr>
<td><strong>Sum(IEnumerable&lt;Expression&gt;)</strong></td>
<td>Sums the specified expressions.</td>
</tr>
<tr>
<td><strong>Sum(IEnumerable&lt;Term&gt;)</strong></td>
<td>Sums the specified terms.</td>
</tr>
<tr>
<td><strong>Sum(IEnumerable&lt;Variable&gt;)</strong></td>
<td>Sums the specified variables.</td>
</tr>
<tr>
<td><strong>Sum(Int32, Int32, Func(Int32, Expression))</strong></td>
<td>Sums the specified range of expressions.</td>
</tr>
<tr>
<td><strong>Sum(Int32, Int32, Func(Int32, Term))</strong></td>
<td>Sums the specified range of terms.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="https://docs.microsoft.com/en-us/dotnet/api/system.string">String</a> that represents the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td><strong>Addition(Double, Expression)</strong></td>
<td>Adds a constant to an expression</td>
</tr>
<tr>
<td><strong>Addition(Expression, Expression)</strong></td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td><strong>Addition(Expression, Variable)</strong></td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td><strong>Addition(Expression, Double)</strong></td>
<td>Adds a constant to an expression</td>
</tr>
<tr>
<td><strong>Addition(Variable, Expression)</strong></td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td><strong>Equality(Expression, Expression)</strong></td>
<td>Implements the operator ==.</td>
</tr>
<tr>
<td><strong>Equality(Expression, Double)</strong></td>
<td>Implements the operator ==.</td>
</tr>
<tr>
<td><strong>Equality(Variable, Expression)</strong></td>
<td>Creates a new equality constraint</td>
</tr>
<tr>
<td><strong>GreaterThanOrEqual(Double, Expression)</strong></td>
<td>Implements the operator &gt;=.</td>
</tr>
<tr>
<td><strong>GreaterThanOrEqual(Expression, Expression)</strong></td>
<td>Implements the operator &gt;=.</td>
</tr>
<tr>
<td><strong>GreaterThanOrEqual(Expression, Variable)</strong></td>
<td>Implements the operator &gt;=.</td>
</tr>
<tr>
<td><strong>GreaterThanOrEqual(Expression, Double)</strong></td>
<td>Implements the operator &gt;=.</td>
</tr>
<tr>
<td><strong>GreaterThanOrEqual(Variable, Expression)</strong></td>
<td>Implements the operator &gt;=.</td>
</tr>
<tr>
<td><strong>Inequality(Expression, Expression)</strong></td>
<td>Please construct inequalities using &lt;= and &gt;=</td>
</tr>
<tr>
<td><strong>Inequality(Expression, Variable)</strong></td>
<td>Please construct inequalities using &lt;= and &gt;=</td>
</tr>
<tr>
<td><strong>Inequality(Expression, Double)</strong></td>
<td>Please construct inequalities using &lt;= and &gt;=</td>
</tr>
<tr>
<td><strong>Inequality(Variable, Expression)</strong></td>
<td>Please construct inequalities using &lt;= and &gt;=</td>
</tr>
<tr>
<td><strong>LessThanOrEqual(Double, Expression)</strong></td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td>Expression Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>LessThanOrEqual(Expression,</td>
<td></td>
</tr>
<tr>
<td>Expression)</td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td>LessThanOrEqual(Expression,</td>
<td></td>
</tr>
<tr>
<td>Variable)</td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td>LessThanOrEqual(Expression,</td>
<td></td>
</tr>
<tr>
<td>Double)</td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td>LessThanOrEqual(Variable,</td>
<td></td>
</tr>
<tr>
<td>Expression)</td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td>Multiply(Double, Expression)</td>
<td>Implements the operator * .</td>
</tr>
<tr>
<td>Multiply(Expression,</td>
<td></td>
</tr>
<tr>
<td>Expression)</td>
<td>Implements the operator * .</td>
</tr>
<tr>
<td>Multiply(Expression, Double)</td>
<td>Implements the operator * .</td>
</tr>
<tr>
<td>Subtraction(Double,</td>
<td></td>
</tr>
<tr>
<td>Expression)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Expression,</td>
<td></td>
</tr>
<tr>
<td>Expression)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Expression,</td>
<td></td>
</tr>
<tr>
<td>Variable)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Expression,</td>
<td></td>
</tr>
<tr>
<td>Double)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Variable,</td>
<td></td>
</tr>
<tr>
<td>Expression)</td>
<td>Implements the operator -.</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Constant</strong></td>
<td>Gets the constant of this expression.</td>
</tr>
<tr>
<td><strong>ExpressionLowerEstimate</strong></td>
<td>Gives a lower estimate on the value of an Expression. ToDo: Very experimental. Should be able to estimate linear combinations of variables with finite bounds. In the default case -infinity is returned!</td>
</tr>
<tr>
<td><strong>ExpressionUpperEstimate</strong></td>
<td>Gives an upper estimate in the value of an Expression. ToDo: Very experimental. Should be able to estimate linear combinations of variables with finite bounds. In the default case -infinity is returned!</td>
</tr>
<tr>
<td><strong>isLinear</strong></td>
<td>Gets or sets a value indicating whether this instance is linear.</td>
</tr>
<tr>
<td><strong>Terms</strong></td>
<td>Gets a list of the terms in this expression. It does not contain any operators and does not tell you anything about how they are combined with operators.</td>
</tr>
<tr>
<td><strong>Variables</strong></td>
<td>Gets the variables in this expression.</td>
</tr>
</tbody>
</table>
See Also

Expression Class
Optimization Namespace
Expression Constructor

Initializes a new instance of the Expression class

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>protected Expression()</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Protected Sub New</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>protected: Expression()</td>
</tr>
</tbody>
</table>
See Also

Expression Class
Optimization Namespace
The Expression type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>Evaluate</strong></td>
<td>Evaluates the expression using the specified variable values.</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>Sum(IEnumerable&lt;Double&gt;)</strong></td>
<td>Sums up the specified values.</td>
</tr>
<tr>
<td><strong>Sum(IEnumerable&lt;Expression&gt;)</strong></td>
<td>Sums the specified expressions.</td>
</tr>
<tr>
<td><strong>Sum(IEnumerable&lt;Term&gt;)</strong></td>
<td>Sums the specified terms.</td>
</tr>
<tr>
<td><strong>Sum(IEnumerable&lt;Variable&gt;)</strong></td>
<td>Sums the specified variables.</td>
</tr>
<tr>
<td><strong>Sum(Int32, Int32, Func(Int32, Expression))</strong></td>
<td>Sums the specified range of expressions.</td>
</tr>
<tr>
<td><strong>Sum(Int32, Int32, Func(Int32, Term))</strong></td>
<td>Sums the specified range of terms.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
</tbody>
</table>
See Also

Expression Class
Optimization Namespace
Evaluates the expression using the specified variable values.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public double Evaluate(
    IDictionary<string, double> variableValues
)
```

Visual Basic

```vbnet
Public Function Evaluate ( _
    variableValues As IDictionary(Of String, Double)
) As Double
```

Visual C++

```cpp
public:
    double Evaluate(
        IDictionary<String^, Double>^ variableValues
    )
```

Parameters

variableValues
Type: `System.Collections.Generic.IDictionary(String, Double)`
The variable values.

Return Value

See Also

Expression Class
Optimization Namespace
Optimization Framework

Expression.Sum Method

Expression Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sum(IEnumerable&lt;Double&gt;())</td>
<td>Sums up the specified values.</td>
</tr>
<tr>
<td>Sum(IEnumerable&lt;Expression&gt;())</td>
<td>Sums the specified expressions.</td>
</tr>
<tr>
<td>Sum(IEnumerable&lt;Term&gt;())</td>
<td>Sums the specified terms.</td>
</tr>
<tr>
<td>Sum(IEnumerable&lt;Variable&gt;())</td>
<td>Sums the specified variables.</td>
</tr>
<tr>
<td>Sum(Int32, Int32, Func(Int32, Expression))</td>
<td>Sums the specified range of expressions.</td>
</tr>
<tr>
<td>Sum(Int32, Int32, Func(Int32, Term))</td>
<td>Sums the specified range of terms.</td>
</tr>
</tbody>
</table>
See Also

Expression Class
Expression Members
Optimization Namespace
Sums up the specified values.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Expression Sum(
    IEnumerable<double> values
)
```

**Visual Basic**

```vbnet
Public Shared Function Sum ( _
    values As IEnumerable(Of Double) _
) As Expression
```

**Visual C++**

```cpp
public:
static Expression^ Sum(
    IEnumerable<double>^ values
)
```

### Parameters

`values`

Type: `System.Collections.Generic.IEnumerable<Double>`

The expressions.

### Return Value

[Missing <returns> documentation for
# Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>values != null</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;Expression&gt;() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Expression Class
Sum Overload
Optimization Namespace
Sums the specified expressions.

**Namespace:** [Optimization](http://example.com)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public static Expression Sum(
    IEnumerable<Expression> expressions
)
```

Visual Basic

```vbnet
Public Shared Function Sum (_
    expressions As IEnumerable(Of Expression) _
) As Expression
```

Visual C++

```cpp
public:
static Expression^ Sum(
    IEnumerable<Expression^>^ expressions
)
```

Parameters

expressions
Type: `System.Collections.Generic.IEnumerable<Expression>
The expressions.

Return Value

[Missing <returns> documentation for
Expressions)"
]
# Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>expressions != null</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;Expression&gt;() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Expression Class
Sum Overload
Optimization Namespace
Sums the specified terms.

**Namespace:** [Optimization](https://example.com)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Expression Sum(
    IEnumerable<Term> expressions
)
```

**Visual Basic**

```vbnet
Public Shared Function Sum ( _
    expressions As IEnumerable(Of Term) _
) As Expression
```

**Visual C++**

```cpp
public:
    static Expression^ Sum(
        IEnumerable<Term^>^ expressions
    )
```

### Parameters

*expressions*

Type: `System.Collections.Generic.IEnumerable<Term>`

The terms.

### Return Value

[Missing <returns> documentation for
See Also

Expression Class
Sum Overload
Optimization Namespace
Sums the specified variables.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Expression Sum(
    IEnumerable<Variable> variables
)
```

**Visual Basic**

```vbnet
Public Shared Function Sum ( _
    variables As IEnumerable(Of Variable) _
) As Expression
```

**Visual C++**

```cpp
public:
static Expression^ Sum(
    IEnumerable<Variable^>^ variables
)
```

### Parameters

**variables**

Type: `System.Collections.Generic.IEnumerable<Variable>`

The variables.

### Return Value

[Missing <returns> documentation for
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>variables != null</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;Expression&gt;() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Expression Class
Sum Overload
Optimization Namespace
Sums the specified range of expressions.

**Namespace:** [Optimization](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public static Expression Sum(
    int start,
    int count,
    Func<int, Expression> selector
)
```

Visual Basic

```vbnet
Public Shared Function Sum ( _
    start As Integer, _
    count As Integer, _
    selector As Func(Of Integer, Expression) _
) As Expression
```

Visual C++

```csharp
public:
static Expression^ Sum(
    int start,
    int count,
    Func<int, Expression^>^ selector
)
```

Parameters

start
Type: System.Int32
The start.

count
Type: System.Int32
The count.
selector
Type: System.Func(Int32, Expression)
The selector.

Return Value
### Requires

<table>
<thead>
<tr>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>selector != null</td>
</tr>
<tr>
<td>count &gt;= 0</td>
</tr>
<tr>
<td>start &gt;= 0</td>
</tr>
<tr>
<td>start + count - 1 &lt;= int.MaxValue</td>
</tr>
</tbody>
</table>

### Ensures

(Object)Contract.Result<Expression>() != null

[Learn more about contracts](#)
See Also

Expression Class
Sum Overload
Optimization Namespace
Sums the specified range of terms.

**Namespace**: Optimization

**Assembly**: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Expression Sum(
    int start,
    int count,
    Func<int, Term> selector
)
```

**Visual Basic**

```vbnet
Public Shared Function Sum (
    start As Integer, _
    count As Integer, _
    selector As Func(Of Integer, Term) _
) As Expression
```

**Visual C++**

```cpp
public:
static Expression^ Sum(
    int start,
    int count,
    Func<int, Term^>^ selector
)
```

### Parameters

**start**
Type: System.Int32
The start.

**count**
Type: System.Int32
The count.
selector
Type: `System.Func<Int32, Term>`
The selector.

**Return Value**

Contracts

**Requires**

- `selector != null`
- `count >= 0`
- `start >= 0`
- `start + count - 1 <= int.MaxValue`

**Ensures**

- `(Object)Contract.Result<Expression>() != null`

[Learn more about contracts](#)
See Also

Expression Class
Sum Overload
Optimization Namespace
Optimization Framework

Expression Operators

Expression Class  See Also  Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addition(Double, Expression)</td>
<td>Adds a constant to an expression</td>
</tr>
<tr>
<td>Addition(Expression, Expression)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Expression, Variable)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Expression, Double)</td>
<td>Adds a constant to an expression</td>
</tr>
<tr>
<td>Addition(Variable, Expression)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Equality(Expression, Expression)</td>
<td>Implements the operator ==.</td>
</tr>
<tr>
<td>Equality(Expression, Variable)</td>
<td>Implements the operator ==.</td>
</tr>
<tr>
<td>Equality(Expression, Double)</td>
<td>Implements the operator ==.</td>
</tr>
<tr>
<td>Equality(Variable, Expression)</td>
<td>Creates a new equality constraint</td>
</tr>
<tr>
<td>GreaterThanOrEqual(Double, Expression)</td>
<td>Implements the operator &gt;=.</td>
</tr>
<tr>
<td>GreaterThanOrEqual(Expression, Expression)</td>
<td>Implements the operator &gt;=.</td>
</tr>
<tr>
<td>GreaterThanOrEqual(Expression, Variable)</td>
<td>Implements the operator &gt;=.</td>
</tr>
<tr>
<td>GreaterThanOrEqual(Expression, Double)</td>
<td>Implements the operator &gt;=.</td>
</tr>
<tr>
<td>GreaterThanOrEqual(Variable, Expression)</td>
<td>Implements the operator &gt;=.</td>
</tr>
<tr>
<td>Inequality(Expression, Expression)</td>
<td>Please construct inequalities using &lt;= and &gt;=</td>
</tr>
<tr>
<td>Inequality(Expression, Variable)</td>
<td>Please construct inequalities using &lt;= and &gt;=</td>
</tr>
<tr>
<td>Inequality(Expression, Double)</td>
<td>Please construct inequalities using &lt;= and &gt;=</td>
</tr>
<tr>
<td>Inequality(Variable, Expression)</td>
<td>Please construct inequalities using &lt;= and &gt;=</td>
</tr>
<tr>
<td>LessThanOrEqual(Double, Expression)</td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td>Expression</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>---------------------------------------</td>
</tr>
<tr>
<td><code>LessThanOrEqual(Expression, Expression)</code></td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td><code>LessThanOrEqual(Expression, Variable)</code></td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td><code>LessThanOrEqual(Expression, Double)</code></td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td><code>LessThanOrEqual(Variable, Expression)</code></td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td><code>Multiply(Double, Expression)</code></td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td><code>Multiply(Expression, Expression)</code></td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td><code>Multiply(Expression, Double)</code></td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td><code>Subtraction(Double, Expression)</code></td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td><code>Subtraction(Expression, Expression)</code></td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td><code>Subtraction(Expression, Variable)</code></td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td><code>Subtraction(Expression, Double)</code></td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td><code>Subtraction(Variable, Expression)</code></td>
<td>Implements the operator -.</td>
</tr>
</tbody>
</table>
See Also

Expression Class
Optimization Namespace
Optimization Framework

Expression.Addition Operator

Expression Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addition(Double, Expression)</td>
<td>Adds a constant to an expression</td>
</tr>
<tr>
<td>Addition(Expression, Expression)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Expression, Variable)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Expression, Double)</td>
<td>Adds a constant to an expression</td>
</tr>
<tr>
<td>Addition(Variable, Expression)</td>
<td>Implements the operator +.</td>
</tr>
</tbody>
</table>
See Also

Expression Class
Expression Members
Optimization Namespace
Expression.Addition Operator (Double, Expression)

Adding a constant to an expression

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Expression operator +(double constant, Expression expression)
```

**Visual Basic**

```vbnet
Public Shared Operator + (constant As Double, expression As Expression) As Expression
```

**Visual C++**

```cpp
public:
static Expression^ operator +(double constant, Expression^ expression)
```

### Parameters

- **constant**  
  Type: `System.Double`  
  [Missing <param name="constant"/> documentation for "M:Optimization.Expression.op_Addition(System.Double,Optimization.Expression")

- **expression**  
  Type: `Optimization.Expression`  
  [Missing <param name="expression"/> documentation for "M:Optimization.Expression.op_Addition(System.Double,Optimization.Expression")
Return Value

[Missing <returns> documentation for "M:Optimization.Expression.op_Addition(System.Double,Optimization.Expression)"
<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)expression != null</td>
</tr>
<tr>
<td>Ensures</td>
</tr>
<tr>
<td>(Object)Contract.Result&lt;Expression&gt;() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Expression Class
Addition Overload
Optimization Namespace
Implement the operator +.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>public static Expression operator + ( Expression expression1, Expression expression2 )</td>
</tr>
<tr>
<td>Visual Basic</td>
<td>Public Shared Operator + ( _ expression1 As Expression, _ expression2 As Expression _ ) As Expression</td>
</tr>
<tr>
<td>Visual C++</td>
<td>public: static Expression^ operator +( Expression^ expression1, Expression^ expression2 )</td>
</tr>
</tbody>
</table>

### Parameters

- **expression1**
  - Type: Optimization.Expression
  - The first expression.

- **expression2**
  - Type: Optimization.Expression
  - The second expression.

### Return Value
The result of the operator.
<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!object.Equals(expression1, null)</td>
</tr>
<tr>
<td>!object.Equals(expression2, null)</td>
</tr>
<tr>
<td>Ensures</td>
</tr>
<tr>
<td>(Object)Contract.Result&lt;Expression&gt;() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Expression Class
Addition Overload
Optimization Namespace
Optimization Framework

Expression.Addition Operator (Expression, Variable)

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0

Implements the operator +.
Syntax

C#

```csharp
public static Expression operator +(Expression expression1,
    Variable variable)
```

Visual Basic

```vbnet
Public Shared Operator + ( _
    expression1 As Expression, _
    variable As Variable _
) As Expression
```

Visual C++

```cpp
public:
static Expression^ operator +(Expression^ expression1,
    Variable^ variable)
```

Parameters

ddexpression1
Type: Optimization.Expression
The expression.

ddvariable
Type: Optimization.Variable
The variable.

Return Value
The result of the operator.
## Contracts

### Requires

- `(Object)expression1 != null`
- `(Object)variable != null`

### Ensures

- `(Object)Contract.Result<Expression>() != null`

[Learn more about contracts]
See Also

Expression Class
Addition Overload
Optimization Namespace
Optimization Framework

Expression.Addition Operator (Expression, Double)

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0

Adds a constant to an expression
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td><code>public static Expression operator +(Expression expression, double constant)</code></td>
</tr>
</tbody>
</table>
| Visual Basic | `Public Shared Operator + ( _
expression As Expression, _
constant As Double _
) As Expression` |
| Visual C++ | `public:
static Expression^ operator +(Expression^ expression,
double constant)` |

### Parameters

**expression**
Type: `Optimization.Expression`

**constant**
Type: `System.Double`
Return Value

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)expression != null</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;Expression&gt;() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Expression Class
Addition Overload
Optimization Namespace
Optimization Framework

Expression/Addition Operator (Variable, Expression)

Implement the operator +.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

**C#**

```csharp
public static Expression operator +(Variable variable,
    Expression expression1
)
```

**Visual Basic**

```vbnet
Public Shared Operator +( _
    variable As Variable,
    expression1 As Expression_
) As Expression
```

**Visual C++**

```cpp
public:
static Expression^ operator +(Variable^ variable,
    Expression^ expression1
)
```

**Parameters**

- **variable**
  Type: `Optimization,Variable`
  The variable.

- **expression1**
  Type: `Optimization,Expression`
  The expression.

**Return Value**
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>variable != null</td>
</tr>
<tr>
<td>(Object)expression1 != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Expression Class
Addition Overload
Optimization Namespace
Optimization Framework

Expression.Equality Operator

Expression Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equality(Expression, Expression)</td>
<td>Implements the operator (==).</td>
</tr>
<tr>
<td>Equality(Expression, Variable)</td>
<td></td>
</tr>
<tr>
<td>Equality(Expression, Double)</td>
<td>Implements the operator (==).</td>
</tr>
<tr>
<td>Equality(Variable, Expression)</td>
<td>Creates a new equality constraint</td>
</tr>
</tbody>
</table>
See Also

Expression Class
Expression Members
Optimization Namespace
Implementation of the operator `==`.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Constraint operator ==(
    Expression expression1,
    Expression expression2
)
```

**Visual Basic**

```vbnet
Public Shared Operator = ( _
    expression1 As Expression, _
    expression2 As Expression _
) As Constraint
```

**Visual C++**

```cpp
public:
static Constraint^ operator ==(
    Expression^ expression1,
    Expression^ expression2
)
```

### Parameters

- **expression1**
  - Type: `Optimization.Expression`
  - The first expression.

- **expression2**
  - Type: `Optimization.Expression`
  - The second expression.

### Return Value
The result of the operator.
## Contracts

**Requires**

!object.Equals(expression1, null)

!object.Equals(expression2, null)

**Ensures**

Contract.Result<Constraint>() != null

[Learn more about contracts](#)
See Also

Expression Class
Equality Overload
Optimization Namespace
Expression.Equality Operator (Expression, Variable)

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
C#  

```csharp
public static Constraint operator ==(
    Expression expression,
    Variable variable
)
```

Visual Basic  

```vbnet
Public Shared Operator = ( _
    expression As Expression, _
    variable As Variable _
) As Constraint
```

Visual C++  

```cpp
public:
static Constraint^ operator ==(
    Expression^ expression,
    Variable^ variable
)
```

**Parameters**

*expression*
Type: `Optimization.Expression`

*variable*
Type: `Optimization.Variable`

[Missing <param name="expression"/> documentation for "M:Optimization.Expression.op_Equality(Optimization.Expression,Optim:"

[Missing <param name="variable"/> documentation for "M:Optimization.Expression.op_Equality(Optimization.Expression,Optim:"

Return Value

## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!object.Equals(expression, null)</td>
</tr>
<tr>
<td>!object.Equals(variable, null)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contract.Result&lt;Constraint&gt;() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Expression Class
Equality Overload
Optimization Namespace
Expression.Equality Operator (Expression, Double)

Implements the operator `==`.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

**C#**

```csharp
public static Constraint operator ==(
    Expression expression,
    double value
)
```

**Visual Basic**

```vbnet
Public Shared Operator = ( _
    expression As Expression, _
    value As Double _
) As Constraint
```

**Visual C++**

```cpp
public:
static Constraint^ operator ==(
    Expression^ expression,
    double value
)
```

**Parameters**

*expression*
Type: `Optimization.Expression`
The expression.

*value*
Type: `System.Double`
The constant.

**Return Value**
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!object.Equals(expression, null)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contract.Result&lt;Constraint&gt;() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Expression Class
Equality Overload
Optimization Namespace
Optimization Framework

Expression.Equality Operator (Variable, Expression)

Creates a new equality constraint

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td><code>public static Constraint operator ==( Variable variable, Expression expression)</code></td>
</tr>
<tr>
<td>Visual Basic</td>
<td><code>Public Shared Operator = ( _ variable As Variable, _ expression As Expression _ ) As Constraint</code></td>
</tr>
<tr>
<td>Visual C++</td>
<td><code>public: static Constraint^ operator ==( Variable^ variable, Expression^ expression)</code></td>
</tr>
</tbody>
</table>

**Parameters**

- **variable**
  Type: `Optimization,Variable`
  The variable.

- **expression**
  Type: `Optimization,Expression`
  The expression.

**Return Value**
The a new constraint as the result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!object.Equals(expression, null)</td>
</tr>
<tr>
<td>!object.Equals(variable, null)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contract.Result&lt;Constraint&gt;() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Expression Class
Equality Overload
Optimization Namespace
Optimization Framework

**Expression.GreaterThanOrEqual Operator**

[Expression Class See Also Send Feedback]
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GreaterThanOrEqual(Double, Expression)</td>
<td>Implements the operator &gt;=.</td>
</tr>
<tr>
<td>GreaterThanOrEqual(Expression, Expression)</td>
<td>Implements the operator &gt;=.</td>
</tr>
<tr>
<td>GreaterThanOrEqual(Expression, Variable)</td>
<td>Implements the operator &gt;=.</td>
</tr>
<tr>
<td>GreaterThanOrEqual(Expression, Double)</td>
<td>Implements the operator &gt;=.</td>
</tr>
<tr>
<td>GreaterThanOrEqual(Variable, Expression)</td>
<td>Implements the operator &gt;=.</td>
</tr>
</tbody>
</table>
See Also

Expression Class
Expression Members
Optimization Namespace
Expression.GreaterThanOrEqual Operator (Double, Expression)

Implements the operator >=.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Constraint operator >=(
    double constant,
    Expression expression
)
```

**Visual Basic**

```vbnet
Public Shared Operator >= ( _
    constant As Double, _
    expression As Expression _
) As Constraint
```

**Visual C++**

```cpp
public:
static Constraint^ operator >=(
    double constant,
    Expression^ expression
)
```

### Parameters

*constant*

Type: System.Double

The constant.

*expression*

Type: Optimization.Expression

The expression.

### Return Value
The result of the operator.
# Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!object.Equals(expression, null)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contract.Result&lt;Constraint&gt;() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Expression Class
GreaterThanOrEqual Overload
Optimization Namespace
Optimization Framework

Expression.GreaterThanOrEqual Operator (Expression, Expression)

Implements the operator >=.

Namespace: Optimization

Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```
public static Constraint operator >=(
    Expression expression1,
    Expression expression2
)
```

### Visual Basic

```
Public Shared Operator >= (_
    expression1 As Expression, _
    expression2 As Expression _
) As Constraint
```

### Visual C++

```
public:
static Constraint^ operator >=(
    Expression^ expression1,
    Expression^ expression2
)
```

### Parameters

- **expression1**
  Type: `Optimization.Expression`
  The first expression.

- **expression2**
  Type: `Optimization.Expression`
  The second expression.

### Return Value
The result of the operator.
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!object.Equals(expression1, null)</td>
</tr>
<tr>
<td>!object.Equals(expression2, null)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contract.Result&lt;Constraint&gt;() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Expression Class
GreaterThanOrEqual Overload
Optimization Namespace
Implements the operator \( \geq \).

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

**C#**

```csharp
public static Constraint operator >=(
    Expression expression,
    Variable variable
)
```

**Visual Basic**

```vbs
Public Shared Operator >= ( _
    expression As Expression, _
    variable As Variable _
) As Constraint
```

**Visual C++**

```cpp
public:
static Constraint^ operator >=(
    Expression^ expression, _
    Variable^ variable
)
```

**Parameters**

*expression*
Type: **Optimization.Expression**
The expression.

*variable*
Type: **Optimization.Variable**
The variable.

**Return Value**
The result of the operator.
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!object.Equals(expression, null)</td>
</tr>
<tr>
<td>!object.Equals(variable, null)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contract.Result&lt;Constraint&gt;() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Expression Class
GreaterThanOrEqual Overload
Optimization Namespace
Implements the operator >=.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

<table>
<thead>
<tr>
<th>Programming Language</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public static Constraint operator &gt;= (Expression expression, double constant)</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Shared Operator &gt;= ( _expression As Expression, _constant As Double _ ) As Constraint</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: static Constraint^ operator &gt;= (Expression^ expression, double constant)</td>
</tr>
</tbody>
</table>

**Parameters**

*expression*

Type: **Optimization.Expression**

The expression.

*constant*

Type: **System.Double**

The constant.

**Return Value**
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!object.Equals(expression, null)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contract.Result&lt;Constraint&gt;() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Expression Class
GreaterThanOrEqual Overload
Optimization Namespace
Optimization Framework

Expression.GreaterThanOrEqual Operator (Variable, Expression)

Implements the operator >=.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public static Constraint operator >=(
    Variable variable,
    Expression expression
)
```

Visual Basic

```vbnet
Public Shared Operator >= ( _
    variable As Variable, _
    expression As Expression _
) As Constraint
```

Visual C++

```cpp
public:
static Constraint^ operator >=(
    Variable^ variable,
    Expression^ expression
)
```

Parameters

- **variable**
  Type: **Optimization.Variable**
  The variable.

- **expression**
  Type: **Optimization.Expression**
  The expression.

Return Value
The result of the operator.
## Contracts

### Requires


!object.Equals(expression, null)

!object.Equals(variable, null)

### Ensures


Contract.Result<Constraint>() != null

[Learn more about contracts](#)
See Also

Expression Class
GreaterThanOrEqual Overload
Optimization Namespace
Optimization Framework

Expression.Inequality Operator

Expression Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Inequality(Expression, Expression)</code></td>
<td>Please construct inequalities using &lt;= and &gt;=</td>
</tr>
<tr>
<td><code>Inequality(Expression, Variable)</code></td>
<td>Please construct inequalities using &lt;= and &gt;=</td>
</tr>
<tr>
<td><code>Inequality(Expression, Double)</code></td>
<td>Please construct inequalities using &lt;= and &gt;=</td>
</tr>
<tr>
<td><code>Inequality(Variable, Expression)</code></td>
<td>Please construct inequalities using &lt;= and &gt;=</td>
</tr>
</tbody>
</table>
See Also

Expression Class
Expression Members
Optimization Namespace
Optimization Framework

Expression.Inequality Operator (Expression, Expression)

See Also Send Feedback

Please construct inequalities using <= and >=

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Constraint operator !=(
    Expression expression,
    Expression expression2
)
```

**Visual Basic**

```vbnet
Public Shared Operator <> ( _
    expression As Expression, _
    expression2 As Expression _
) As Constraint
```

**Visual C++**

```cpp
public:
static Constraint^ operator !=(
    Expression^ expression, 
    Expression^ expression2
)
```

### Parameters

- **expression**
  - Type: `Optimization.Expression`
  - The expression.

- **expression2**
  - Type: `Optimization.Expression`
  - The expression2.

### Return Value
The result of the operator.
See Also

Expression Class
Inequality Overload
Optimization Namespace
Please construct inequalities using <= and >=

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>public static Constraint operator !=( Expression expression, Variable variable )</td>
</tr>
<tr>
<td>Visual Basic</td>
<td>Public Shared Operator &lt;&gt; ( _ expression As Expression, _ variable As Variable _ ) As Constraint</td>
</tr>
<tr>
<td>Visual C++</td>
<td>public: static Constraint^ operator !=( Expression^ expression, Variable^ variable )</td>
</tr>
</tbody>
</table>

### Parameters

- **expression**
  - Type: Optimization.Expression
  - The expression.

- **variable**
  - Type: Optimization.Variable
  - The variable.

### Return Value
The result of the operator.
See Also

Expression Class
Inequality Overload
Optimization Namespace
Please construct inequalities using <= and >=

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public static Constraint operator !=(
    Expression expression,
    double value
)
```

### Visual Basic

```vbnet
Public Shared Operator <> ( _
    expression As Expression, _
    value As Double _
) As Constraint
```

### Visual C++

```cpp
public:
static Constraint^ operator !=(
    Expression^ expression,
    double value
)
```

## Parameters

- **expression**
  - Type: `Optimization.Expression`
  - The expression.

- **value**
  - Type: `System.Double`
  - The value.

## Return Value
The result of the operator.
See Also

Expression Class
Inequality Overload
Optimization Namespace
Please construct inequalities using \(\leq\) and \(\geq\)

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>public static Constraint operator !=(Variable variable, Expression expression)</td>
</tr>
</tbody>
</table>
| Visual Basic | Public Shared Operator <> ( _
|             | variable As Variable, _
|             | expression As Expression _ ) As Constraint |
| Visual C++ | public: static Constraint variable expression !=(Variable variable, Expression expression) |

### Parameters

- **variable**
  - Type: `Optimization.Variable`  
The variable.

- **expression**
  - Type: `Optimization.Expression`  
The expression.

### Return Value
The result of the operator.
See Also

Expression Class
Inequality Overload
Optimization Namespace
Optimization Framework

Expression.LessThanOrEqual Operator

Expression Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LessThanOrEqual(Double, Expression)</td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td>LessThanOrEqual(Expression, Expression)</td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td>LessThanOrEqual(Expression, Variable)</td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td>LessThanOrEqual(Expression, Double)</td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td>LessThanOrEqual(Variable, Expression)</td>
<td>Implements the operator &lt;=.</td>
</tr>
</tbody>
</table>
See Also

Expression Class
Expression Members
Optimization Namespace
Expression.LessThanOrEqual Operator (Double, Expression)

Implements the operator <=.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public static Constraint operator <=(
    double constant,
    Expression expression
)
```

### Visual Basic

```vbnet
Public Shared Operator <= ( _
    constant As Double, _
    expression As Expression _
) As Constraint
```

### Visual C++

```cpp
public:
static Constraint^ operator <=(
    double constant,
    Expression^ expression
)
```

## Parameters

- **constant**
  Type: **System.Double**
  The constant.

- **expression**
  Type: **Optimization.Expression**
  The expression.

## Return Value
The result of the operator.
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>!object.Equals(expression, null)</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Contract.Result&lt;Constraint&gt;(() ! null)</code></td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Expression Class
LessThanOrEqual Overload
Optimization Namespace
Implements the operator <=.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Constraint operator <=(
    Expression expression1,
    Expression expression2
)
```

**Visual Basic**

```vbnet
Public Shared Operator <= (
    expression1 As Expression,
    expression2 As Expression
) As Constraint
```

**Visual C++**

```cpp
public:
static Constraint^ operator <=(
    Expression^ expression1,
    Expression^ expression2
)
```

### Parameters

- `expression1`
  - Type: `Optimization.Expression`
  - The first expression.

- `expression2`
  - Type: `Optimization.Expression`
  - The second expression.

### Return Value
The result of the operator.
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>!object.Equals(expression1, null)</code></td>
</tr>
<tr>
<td><code>!object.Equals(expression2, null)</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Contract.Result&lt;Constraint&gt;() != null</code></td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Expression Class
LessThanOrEqual Overload
Optimization Namespace
Implementation of the operator \( \leq \).

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public static Constraint operator <=(
    Expression expression,
    Variable variable
)
```

### Visual Basic

```vbnet
Public Shared Operator <= (_
    expression As Expression, _
    variable As Variable _
) As Constraint
```

### Visual C++

```cpp
public: Constraint^ operator <=(
    Expression^ expression, _
    Variable^ variable
)
```

### Parameters

- **expression**
  Type: Optimization.Expression
  The expression.

- **variable**
  Type: Optimization.Variable
  The variable.

### Return Value
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!object.Equals(expression, null)</td>
</tr>
<tr>
<td>!object.Equals(variable, null)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contract.Result&lt;Constraint&gt;() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Expression Class
LessThanOrEqual Overload
Optimization Namespace
Expression. LessThanOrEqual Operator (Expression, Double)

Implements the operator <=.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

public static Constraint operator <=(
    Expression expression,
    double constant
)

Visual Basic

Public Shared Operator <= ( _
    expression As Expression, _
    constant As Double _
) As Constraint

Visual C++

public:
static Constraint^ operator <=(
    Expression^ expression,
    double constant
)

Parameters

expression
Type: Optimization.Expression
The expression.

constant
Type: System.Double
The constant.

Return Value
The result of the operator.
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>!object.Equals(expression, null)</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Contract.Result&lt;Constraint&gt;() != null</code></td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

- Expression Class
- LessThanOrEqual Overload
- Optimization Namespace
Optimization Framework

Expression.LessThanOrEqual Operator (Variable, Expression)

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0

Implements the operator <=.
## Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>public static Constraint operator &lt;=(</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>Variable variable,</code></td>
<td><code>Public Shared Operator &lt;= ( </code>_</td>
<td></td>
</tr>
<tr>
<td><code>Expression expression</code></td>
<td><code>variable As Variable, </code>_</td>
<td></td>
</tr>
<tr>
<td><code>)</code></td>
<td><code>expression As Expression </code>_</td>
<td></td>
</tr>
<tr>
<td></td>
<td>As Constraint</td>
<td></td>
</tr>
</tbody>
</table>

### Parameters

- **variable**
  - Type: `Optimization.Variable`
  - The variable.

- **expression**
  - Type: `Optimization.Expression`
  - The expression.

### Return Value
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!object.Equals(expression, null)</td>
</tr>
<tr>
<td>!object.Equals(variable, null)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contract.Result&lt;Constraint&gt;() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Expression Class
LessThanOrEqual Overload
Optimization Namespace
Optimization Framework

**Expression.Multiply Operator**

[Expression Class] [See Also] [Send Feedback]
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiply(Double, Expression)</td>
<td>Implements the operator <em>.</em></td>
</tr>
<tr>
<td>Multiply(Expression, Expression)</td>
<td>Implements the operator <em>.</em></td>
</tr>
<tr>
<td>Multiply(Expression, Double)</td>
<td>Implements the operator <em>.</em></td>
</tr>
</tbody>
</table>
See Also

Expression Class
Expression Members
Optimization Namespace
Implement the operator `*`.

**Namespace**: Optimization

**Assembly**: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Expression operator *(  
    double constant,  
    Expression expression  
)  
```

**Visual Basic**

```vbnet
Public Shared Operator * (  _
    constant As Double, _
    expression As Expression _
) As Expression  
```

**Visual C++**

```cpp
public:
static Expression^ operator *(  
    double constant,  
    Expression^ expression  
)  
```

### Parameters

- **constant**
  - Type: `System.Double`
  - The constant.

- **expression**
  - Type: `Optimization.Expression`
  - The expression.

### Return Value
The result of the operator.
Contracts

Requires

!object.Equals(expression, null)

Learn more about contracts
See Also

Expression Class
Multiply Overload
Optimization Namespace
Implements the operator `*`.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Expression operator *(Expression expression, Expression expression2)
```

**Visual Basic**

```vbnet
Public Shared Operator * ( _
    expression As Expression, _
    expression2 As Expression _
) As Expression
```

**Visual C++**

```cpp
public:
static Expression^ operator *(Expression^ expression, Expression^ expression2)
```

### Parameters

- **expression**
  Type: `Optimization.Expression`
  The first expression.

- **expression2**
  Type: `Optimization.Expression`
  The second expression.

### Return Value
The result of the operator.
## Contracts

**Requires**

| (Object)expression != null  
| (Object)expression2 != null |

**Ensures**

| (Object)Contract.Result<Expression>() != null |

[Learn more about contracts](#)
See Also

Expression Class
Multiply Overload
Optimization Namespace
Implements the operator \(*\).

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Expression operator *(Expression expression, double constant)
```

**Visual Basic**

```vbnet
Public Shared Operator *( _
    expression As Expression, _
    constant As Double _
) As Expression
```

**Visual C++**

```cpp
public: 
static Expression^ operator *(Expression^ expression, 
    double constant
)
```

---

### Parameters

- **expression**
  - Type: `Optimization.Expression`
  - The expression.

- **constant**
  - Type: `System.Double`
  - The constant.

### Return Value
The result of the operator.
<table>
<thead>
<tr>
<th>Contracts</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Requires</strong></td>
</tr>
<tr>
<td><code>!object.Equals(expression, null)</code></td>
</tr>
<tr>
<td><strong>Ensures</strong></td>
</tr>
<tr>
<td><code>(Object)Contract.Result&lt;Expression&gt;() != null</code></td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Expression Class
Multiply Overload
Optimization Namespace
Optimization Framework

Expression.Subtraction Operator

Expression Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subtraction(Double, Expression)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Expression, Expression)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Expression, Variable)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Expression, Double)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Variable, Expression)</td>
<td>Implements the operator -.</td>
</tr>
</tbody>
</table>
See Also

Expression Class
Expression Members
Optimization Namespace
Optimization Framework

Expression.Subtraction Operator (Double, Expression)

Expression Class See Also Send Feedback

Implements the operator -.  

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public static Expression operator -(double constant, Expression expression1)
```

### Visual Basic

```vbnet
Public Shared Operator - (constant As Double, expression1 As Expression) As Expression
```

### Visual C++

```cpp
public: static Expression^ operator -(double constant, Expression^ expression1)
```

### Parameters

**constant**
Type: `System.Double`
The constant.

**expression1**
Type: `Optimization::Expression`
The expression1.

### Return Value
The result of the operator.
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>(Object)expression1 != null</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>(Object)Contract.Result&lt;Expression&gt;() != null</code></td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Expression Class
Subtraction Overload
Optimization Namespace
Optimization Framework

Expression.Subtraction Operator (Expression, Expression)

Expression Class  See Also  Send Feedback

Implements the operator -.

Namespace: Optimization

Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Expression operator -(Expression expression1, Expression expression2)
```

**Visual Basic**

```vbnet
Public Shared Operator -(__expression1 As Expression, __expression2 As Expression__) As Expression
```

**Visual C++**

```cpp
public:
static Expression^ operator -(Expression^ expression1, Expression^ expression2)
```

### Parameters

- **expression1**
  - Type: `Optimization.Expression`
  - The first expression.

- **expression2**
  - Type: `Optimization.Expression`
  - The second expression.

### Return Value
The result of the operator.
# Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>!expression1.Equals(null)</code></td>
</tr>
<tr>
<td><code>!expression2.Equals(null)</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>(Object)Contract.Result&lt;Expression&gt;() != null</code></td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Expression Class
Subtraction Overload
Optimization Namespace
Expression.Subtraction Operator (Expression, Variable)

Implements the operator \(-\).

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

### C#

```
public static Expression operator -(  
    Expression expression,   
    Variable variable  
)
```

### Visual Basic

```
Public Shared Operator -(  
    expression As Expression,  
    variable As Variable  
) As Expression
```

### Visual C++

```
public:  
static Expression^ operator -(  
    Expression^ expression,   
    Variable^ variable  
)
```

**Parameters**

- **expression**
  Type: `Optimization.Expression`
  The expression.

- **variable**
  Type: `Optimization.Variable`
  The variable.

**Return Value**
The result of the operator.
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!object.Equals(expression, null)</td>
</tr>
<tr>
<td>!object.Equals(variable, null)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;Expression&gt;(() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Expression Class
Subtraction Overload
Optimization Namespace
Expression.Subtraction Operator (Expression, Double)

Implements the operator -.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Expression operator -(Expression expression1, double constant)
```

**Visual Basic**

```vbnet
Public Shared Operator - (expression1 As Expression, constant As Double) As Expression
```

**Visual C++**

```cpp
public: static Expression^ operator -(Expression^ expression1, double constant)
```

### Parameters

- **expression1**
  - Type: `Optimization::Expression`
  - The expression.

- **constant**
  - Type: `System::Double`
  - The constant.

### Return Value
The result of the operator.
Contracts

**Requires**

(Object) expression1 != null

**Ensures**

(Object) Contract.Result<Expression>() != null

Learn more about contracts
See Also

Expression Class
Subtraction Overload
Optimization Namespace
Optimization Framework

Expression.Subtraction Operator (Variable, Expression)

See Also Send Feedback

Implements the operator -.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

public static Expression operator -(  
    Variable variable,  
    Expression expression  
)

Visual Basic

Public Shared Operator - ( _  
    variable As Variable, _  
    expression As Expression _  
) As Expression

Visual C++

public:  
    static Expression^ operator -(  
    Variable^ variable,  
    Expression^ expression  
)

Parameters

variable
Type: Optimization,Variable
The variable.

expression
Type: Optimization,Expression
The expression.

Return Value
The result of the operator.
### Contracts

**Requires**

- !object.Equals(expression, null)
- !object.Equals(variable, null)

**Ensures**

- (Object)Contract.Result<Expression>() != null

[Learn more about contracts](#)
See Also

Expression Class
Subtraction Overload
Optimization Namespace
The `Expression` type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Constant</strong></td>
<td>Gets the constant of this expression.</td>
</tr>
<tr>
<td><strong>ExpressionLowerEstimate</strong></td>
<td>Gives a lower estimate on the value of an Expression. ToDo: Very experimental. Should be able to estimate linear combinations of variables with finite bounds. In the default case -infinity is returned!</td>
</tr>
<tr>
<td><strong>ExpressionUpperEstimate</strong></td>
<td>Gives an upper estimate in the value of an Expression. ToDo: Very experimental. Should be able to estimate linear combinations of variables with finite bounds. In the default case -infinity is returned!</td>
</tr>
<tr>
<td><strong>isLinear</strong></td>
<td>Gets or sets a value indicating whether this instance is linear.</td>
</tr>
<tr>
<td><strong>Terms</strong></td>
<td>Gets a list of the terms in this expression. It does not contain any operators and does not tell you anything about how they are combined with operators.</td>
</tr>
<tr>
<td><strong>Variables</strong></td>
<td>Gets the variables in this expression.</td>
</tr>
</tbody>
</table>
See Also

Expression Class
Optimization Namespace
Optimization Framework

Expression.Constant Property

Expression Class See Also Send Feedback

Gets the constant of this expression.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
</table>
| ```
public virtual double Constant { get; }
``` | ```
Public Overridable ReadOnly Property Constant As Double
Get
``` | ```
public:
virtual property double Constant {
    double get ();
}
``` |

### Field Value

The constant.
See Also

Expression Class
Optimization Namespace
Gives a lower estimate on the value of an Expression. ToDo: Very experimental. Should be able to estimate linear combinations of variables with finite bounds. In the default case -infinity is returned!

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public <code>double</code> ExpressionLowerEstimate { get; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public ReadOnly Property ExpressionLowerEstimate As Double Get</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: property <code>double</code> ExpressionLowerEstimate { double get (); }</td>
</tr>
</tbody>
</table>
See Also

Expression Class
Optimization Namespace
Gives an upper estimate in the value of an Expression. ToDo: Very experimental. Should be able to estimate linear combinations of variables with finite bounds. In the default case -infinity is returned!

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public <code>double</code> ExpressionUpperEstimate { get; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public ReadOnly Property ExpressionUpperEstimate As Double Get</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: property <code>double</code> ExpressionUpperEstimate { <code>double</code> get (); }</td>
</tr>
</tbody>
</table>
See Also

Expression Class
Optimization Namespace
Gets or sets a value indicating whether this instance is linear.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```
public abstract bool isLinear { get; }
```

**Visual Basic**

```
Public MustOverride ReadOnly Property isLinear As Boolean
Get
```

**Visual C++**

```
public:
virtual property bool isLinear {
    bool get () abstract;
}
```

### Field Value

*true* if this instance is linear; otherwise, *false*. 
See Also

Expression Class
Optimization Namespace
Gets a list of the terms in this expression. It does not contain any operators and does not tell you anything about how they are combined with operators.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public abstract IEnumerable&lt;Term&gt; Terms { get; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public MustOverride ReadOnly Property Terms As IEnumerable&lt;Term&gt;</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: virtual property IEnumerable&lt;Term&gt;^ Terms { IEnumerable&lt;Term&gt;^ get () abstract; }</td>
</tr>
</tbody>
</table>

## Field Value

The terms.

See Also

Expression Class
Optimization Namespace
Gets the variables in this expression.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public abstract IEnumerable<IVariable> Variables { get; }
```

**Visual Basic**

```vbnet
Public MustOverride ReadOnly Property Variables As IEnumerable
Get
```

**Visual C++**

```cpp
public:
virtual property IEnumerable<IVariable ^> Variables;
IEnumerable<IVariable ^> get () abstract;
```

### Field Value

The variables.
See Also

Expression Class
Optimization Namespace
The filetype of a model file

**Namespace:** [Optimization](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public enum FileType</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Enumeration FileType</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public enum class FileType</code></td>
</tr>
</tbody>
</table>
### Members

<table>
<thead>
<tr>
<th>Member name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPS</td>
<td>0</td>
<td>MPS</td>
</tr>
<tr>
<td>OSiL</td>
<td>1</td>
<td>OSiL</td>
</tr>
<tr>
<td>LP</td>
<td>2</td>
<td>LP</td>
</tr>
</tbody>
</table>
See Also

Optimization Namespace
The base class for all generic VariableCollection classes

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public abstract class GenericVariableCollectionBase</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public MustInherit Class GenericVariableCollectionBase</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public ref class GenericVariableCollectionBase abstract</code></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
Optimization.GenericVariableCollectionBase
  Optimization.VariableCollection(T)
  Optimization.VariableCollection(T, T2)
  Optimization.VariableCollection(T, T2, T3)
  Optimization.VariableCollection(T, T2, T3, T4)
  Optimization.VariableCollection(T, T2, T3, T4, T5)
  Optimization.VariableCollection(T, T2, T3, T4, T5, T6)
  Optimization.VariableCollection(T, T2, T3, T4, T5, T6, T7)
  Optimization.VariableCollection(T, T2, T3, T4, T5, T6, T7, T8)
See Also

GenericVariableCollectionBase Members
Optimization Namespace
Optimization Framework

GenericVariableCollectionBase Members

The `GenericVariableCollectionBase` type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>GenericVariableCollectionBase</code></td>
<td>Initializes a new instance of the <code>GenericVariableCollectionBase</code> class</td>
</tr>
</tbody>
</table>
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>_internalVariableCollection</td>
<td></td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index Validation</td>
<td>Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on</td>
</tr>
</tbody>
</table>
See Also

GenericVariableCollectionBase Class
Optimization Namespace
Initializes a new instance of the `GenericVariableCollectionBase` class

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
<tr>
<td><code>protected GenericVariableCollectionBase()</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td></td>
</tr>
<tr>
<td><code>Protected Sub New</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
</tr>
<tr>
<td><code>protected: GenericVariableCollectionBase()</code></td>
<td></td>
</tr>
</tbody>
</table>
See Also

GenericVariableCollectionBase Class
Optimization Namespace
The **GenericVariableCollectionBase** type exposes the following members.
### Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_internalVariableCollection</td>
<td></td>
</tr>
</tbody>
</table>
See Also

GenericVariableCollectionBase Class
Optimization Namespace
[Missing <summary> documentation for
"F:Optimization.GenericVariableCollectionBase._internalVariableCollection"

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version:
0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>protected VariableCollection _internalVariableCollection</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Protected _internalVariableCollection As VariableCollection</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>protected: VariableCollection^ _internalVariableCollection</code></td>
</tr>
</tbody>
</table>
See Also

GenericVariableCollectionBase Class
Optimization Namespace
The `GenericVariableCollectionBase` type exposes the following members.
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

GenericVariableCollectionBase Class
Optimization Namespace
Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public void SetVariableValues(
    IDictionary<string, double> values
)
```

### Visual Basic

```vbnet
Public Sub SetVariableValues ( _
    values As IDictionary(Of String, Double) _
)
```

### Visual C++

```cpp
public:
void SetVariableValues(
    IDictionary<String^, double>^ values
)
```

## Parameters

**values**

Type: `System.Collections.Generic.IDictionary/String, Double`

The values for the variables (e.g. coming from a solution)
See Also

GenericVariableCollectionBase Class
Optimization Namespace
The `GenericVariableCollectionBase` type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index Validation</td>
<td>Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on</td>
</tr>
</tbody>
</table>
See Also

GenericVariableCollectionBase Class
Optimization Namespace
Optimization Framework

**GenericVariableCollectionBase.IndexValidation Property**

*GenericVariableCollectionBase Class  See Also  Send Feedback*

Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on

**Namespace:** [Optimization](https://example.com/)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  **Version:** 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
</tr>
</tbody>
</table>
| ```csharp
public bool IndexValidation { get; set; }
``` |
| **Visual Basic** |
| ```vb
Public Property IndexValidation As Boolean
    Get
    Set
``` |
| **Visual C++** |
| ```cpp
public:
    property bool IndexValidation {
        bool get ();
        void set (bool value);
    }
``` |
See Also

GenericVariableCollectionBase Class
Optimization Namespace
Represents a mathematical model

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

#### C#

```csharp
[SerializableAttribute]
public class Model : ICanRegisterSolvers, ICanTrackPropertyChangesInVariables
```

#### Visual Basic

```vbscript
<SerializableAttribute> _
Public Class Model _
    Implements ICanRegisterSolvers, ICanTrackPropertyChangesInVariables
```

#### Visual C++

```cpp
[SerializableAttribute]
public ref class Model : ICanRegisterSolvers, ICanTrackPropertyChangesInVariables
```
Inheritance Hierarchy

System.Object

Optimization.Model
See Also

Model Members
Optimization Namespace
The `Model` type exposes the following members.
### Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>Initializes a new instance of the Model class.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Abs</code></td>
<td>Returns an Expression object representing the absolute value of the specified expression.</td>
</tr>
<tr>
<td><code>AddAlternativeConstraintGroups</code></td>
<td></td>
</tr>
<tr>
<td><code>AddAlternativeConstraints</code></td>
<td></td>
</tr>
<tr>
<td><code>AddConstraint(Constraint, String)</code></td>
<td>Adds constraint to this model instance if it is consistent. If you choose to provide a name, the name of the constraint will be overridden. You need to make sure that the name of the constraint is unique.</td>
</tr>
<tr>
<td><code>AddConstraint(String, Double, Double, Expression)</code></td>
<td>Adds a new constraint to this model instance if it is consistent.</td>
</tr>
<tr>
<td><code>AddConstraints</code></td>
<td>Adds a set of constraint to this model instance if it is consistent.</td>
</tr>
<tr>
<td><code>AddObjective(Objective)</code></td>
<td>Adds objective to this model instance if it is consistent.</td>
</tr>
<tr>
<td><code>AddObjective(Expression, String, ObjectiveSense)</code></td>
<td>Adds a new objective to this model instance if it is consistent.</td>
</tr>
<tr>
<td><code>AddObjectives</code></td>
<td>Adds a set of objectives to this model instance if it is consistent.</td>
</tr>
<tr>
<td><code>AddSOS</code></td>
<td>Adds an SOS Constraint of the given SOSType</td>
</tr>
<tr>
<td><code>AddSOS1</code></td>
<td>Adds an SOS1 constraint</td>
</tr>
<tr>
<td><code>AddSOS2</code></td>
<td>Adds an SOS2 constraint</td>
</tr>
<tr>
<td><code>AddSOS3</code></td>
<td>Adds an SOS3 constraint</td>
</tr>
<tr>
<td><code>AddVariable(IVariable)</code></td>
<td>Adds variable to this model instance if it is consistent.</td>
</tr>
<tr>
<td><code>AddVariable(String, Double, Double, VariableType)</code></td>
<td>Adds a new variable to this model instance if it is consistent.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>AddVariables</td>
<td>Adds a set of variables to this model instance if it is consistent.</td>
</tr>
<tr>
<td></td>
<td>//TODO: discuss: What do we mean by &quot;consistent&quot;?</td>
</tr>
<tr>
<td>Ceil</td>
<td>Returns an Expression object representing the ceil roundoff of the specified</td>
</tr>
<tr>
<td></td>
<td>expression.</td>
</tr>
<tr>
<td>Clear</td>
<td>Removes all variables, constraints, objectives and scenarios from this model</td>
</tr>
<tr>
<td></td>
<td>instance.</td>
</tr>
<tr>
<td>ContainsConstraint</td>
<td>Contains this model instance a constraint named <code>name</code>?</td>
</tr>
<tr>
<td>ContainsObjective</td>
<td>Contains this model instance an objective named <code>name</code>?</td>
</tr>
<tr>
<td>ContainsVariable</td>
<td>Contains this model instance a variable named <code>name</code>?</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup</td>
</tr>
<tr>
<td></td>
<td>operations before the <code>Object</code> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>Floor</td>
<td>Returns an Expression object representing the floor roundoff of the specified</td>
</tr>
<tr>
<td></td>
<td>expression.</td>
</tr>
<tr>
<td>GetConstraint</td>
<td>Returns the constraint named <code>name</code>, or <code>null</code> if this model instance</td>
</tr>
<tr>
<td></td>
<td>contains no such constraint.</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>GetObjective</td>
<td>Returns the objective named <em>name</em>, or null if this model instance contains no such objective.</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>. )</td>
</tr>
<tr>
<td>GetVariable</td>
<td>Returns the variable named <em>name</em>, or null if this model instance contains no such variable.</td>
</tr>
<tr>
<td>Load</td>
<td>Loads the specified file.</td>
</tr>
<tr>
<td>Max</td>
<td>Returns an Expression object representing the maximum of the specified expressions.</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>. )</td>
</tr>
<tr>
<td>Min</td>
<td>Returns an Expression object representing the minimum of the specified expressions.</td>
</tr>
<tr>
<td>RemoveConstraint</td>
<td>Removes the constraint named <em>name</em> from this model instance, or does nothing if this model instance contains no such constraint.</td>
</tr>
<tr>
<td>RemoveConstraints</td>
<td>Removes a set of constraints from this model instance.</td>
</tr>
<tr>
<td>RemoveObjective</td>
<td>Removes the objective named <em>name</em> from this model instance, or does nothing if this model instance contains no such objective.</td>
</tr>
<tr>
<td>RemoveObjectives</td>
<td>Removes a set of objectives from this model instance.</td>
</tr>
</tbody>
</table>

(Inherited from `Object`.)
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RemoveVariable</strong></td>
<td>Removes the variable named <em>name</em> from this model instance, or does nothing if this model instance contains no such variable.</td>
</tr>
<tr>
<td><strong>RemoveVariables</strong></td>
<td>Removes a set of variables from this model instance.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <em>String</em> that represents the current <em>Object</em>. (Inherited from <em>Object.</em>)</td>
</tr>
<tr>
<td><strong>Write</strong></td>
<td>Writes the model to the specified filestream.</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AreAllConstraintsLinear</td>
<td>Have all constraints in this model instance linear expressions?</td>
</tr>
<tr>
<td>AreAllObjectivesLinear</td>
<td>Have all objectives in this model instance linear expressions?</td>
</tr>
<tr>
<td>AreAllVariablesBinary</td>
<td>Are all variables in this model instance of type integer and their lower bound equal to zero and their upper bound equal to 1?</td>
</tr>
<tr>
<td>AreAllVariablesContinuous</td>
<td>Are variables in this model instance of type continuous?</td>
</tr>
<tr>
<td>AreAllVariablesInteger</td>
<td>Are all variables in this model instance of type integer?</td>
</tr>
<tr>
<td>Constraints</td>
<td>Constraints in this model instance.</td>
</tr>
<tr>
<td>ConstraintsCount</td>
<td>Count of constraints in this model instance.</td>
</tr>
<tr>
<td>ModelBehavior</td>
<td>Gets or sets the behavior of the model. Available options are &quot;Automatic&quot; and &quot;Manual&quot; In automatic mode, the model takes care of adding and removing variables when constraints and objectives get added. In manual mode you need to take care of this yourself.</td>
</tr>
<tr>
<td>Name</td>
<td>Name of this model instance.</td>
</tr>
<tr>
<td>Objectives</td>
<td>Objectives in this model instance.</td>
</tr>
<tr>
<td>ObjectivesCount</td>
<td>Count of objectives in this model instance.</td>
</tr>
<tr>
<td>SOS1Sets</td>
<td>Sets of SOS1 variables in this model instance.</td>
</tr>
<tr>
<td>SOS2Sets</td>
<td>Sets of SOS2 variables in this model instance.</td>
</tr>
<tr>
<td>SOS3Sets</td>
<td>Sets of SOS3 variables in this model instance.</td>
</tr>
<tr>
<td><strong>Variables</strong></td>
<td>Variables in this model instance.</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td><strong>VariablesCount</strong></td>
<td>Count of variables in this model instance.</td>
</tr>
</tbody>
</table>
See Also

Model Class
Optimization Namespace
Initializes a new instance of the Model class.

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>public Model()</td>
</tr>
<tr>
<td>Visual Basic</td>
<td>Public Sub New</td>
</tr>
<tr>
<td>Visual C++</td>
<td>public: Model()</td>
</tr>
</tbody>
</table>
See Also

Model Class
Optimization Namespace
The **Model** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Abs</strong></td>
<td>Returns an Expression object representing the absolute value of the specified expression.</td>
</tr>
<tr>
<td><strong>AddAlternativeConstraintGroups</strong></td>
<td></td>
</tr>
<tr>
<td><strong>AddAlternativeConstraints</strong></td>
<td></td>
</tr>
<tr>
<td><strong>AddConstraint(Constraint, String)</strong></td>
<td>Adds constraint to this model instance if it is consistent. If you choose to provide a name, the name of the constraint will be overridden. You need to make sure that the name of the constraint is unique.</td>
</tr>
<tr>
<td><strong>AddConstraint(String, Double, Double, Expression)</strong></td>
<td>Adds a new constraint to this model instance if it is consistent.</td>
</tr>
<tr>
<td><strong>AddConstraints</strong></td>
<td>Adds a set of constraint to this model instance if it is consistent.</td>
</tr>
<tr>
<td><strong>AddObjective(Objective)</strong></td>
<td>Adds objective to this model instance if it is consistent.</td>
</tr>
<tr>
<td><strong>AddObjective(Expression, String, ObjectiveSense)</strong></td>
<td>Adds a new objective to this model instance if it is consistent.</td>
</tr>
<tr>
<td><strong>AddObjectives</strong></td>
<td>Adds a set of objectives to this model instance if it is consistent.</td>
</tr>
<tr>
<td><strong>AddSOS</strong></td>
<td>Adds an SOS Constraint of the given SOSType</td>
</tr>
<tr>
<td><strong>AddSOS1</strong></td>
<td>Adds an SOS1 constraint</td>
</tr>
<tr>
<td><strong>AddSOS2</strong></td>
<td>Adds an SOS2 constraint</td>
</tr>
<tr>
<td><strong>AddSOS3</strong></td>
<td>Adds an SOS3 constraint</td>
</tr>
<tr>
<td><strong>AddVariable(IVariable)</strong></td>
<td>Adds variable to this model instance if it is consistent.</td>
</tr>
<tr>
<td><strong>AddVariable(String, Double, Double, VariableType)</strong></td>
<td>Adds a new variable to this model instance if it is consistent.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>AddVariables</td>
<td>Adds a set of variables to this model instance if it is consistent. //TODO: discuss: What do we mean by &quot;consistent&quot;?</td>
</tr>
<tr>
<td>Ceil</td>
<td>Returns an Expression object representing the ceil roundoff of the specified expression.</td>
</tr>
<tr>
<td>Clear</td>
<td>Removes all variables, constraints, objectives and scenarios from this model instance.</td>
</tr>
<tr>
<td>ContainsConstraint</td>
<td>Contains this model instance a constraint named name?</td>
</tr>
<tr>
<td>ContainsObjective</td>
<td>Contains this model instance an objective named name?</td>
</tr>
<tr>
<td>ContainsVariable</td>
<td>Contains this model instance a variable named name?</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified Object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td>Floor</td>
<td>Returns an Expression object representing the floor roundoff of the specified expression.</td>
</tr>
<tr>
<td>GetConstraint</td>
<td>Returns the constraint named name, or null if this model instance contains no such constraint.</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>GetObjective</td>
<td>Returns the objective named <em>name</em>, or <em>null</em> if this model instance contains no such objective.</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <em>Type</em> of the current instance. (Inherited from <em>Object.</em>)</td>
</tr>
<tr>
<td>GetVariable</td>
<td>Returns the variable named <em>name</em>, or <em>null</em> if this model instance contains no such variable.</td>
</tr>
<tr>
<td>Load</td>
<td>Loads the specified file.</td>
</tr>
<tr>
<td>Max</td>
<td>Returns an Expression object representing the maximum of the specified expressions.</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <em>Object</em>. (Inherited from <em>Object.</em>)</td>
</tr>
<tr>
<td>Min</td>
<td>Returns an Expression object representing the minimum of the specified expressions.</td>
</tr>
<tr>
<td>RemoveConstraint</td>
<td>Removes the constraint named <em>name</em> from this model instance, or does nothing if this model instance contains no such constraint.</td>
</tr>
<tr>
<td>RemoveConstraints</td>
<td>Removes a set of constraints from this model instance.</td>
</tr>
<tr>
<td>RemoveObjective</td>
<td>Removes the objective named <em>name</em> from this model instance, or does nothing if this model instance contains no such objective.</td>
</tr>
<tr>
<td>RemoveObjectives</td>
<td>Removes a set of objectives from this model instance.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RemoveVariable</td>
<td>Removes the variable named <em>name</em> from this model instance, or does nothing if this model instance contains no such variable.</td>
</tr>
<tr>
<td>RemoveVariables</td>
<td>Removes a set of variables from this model instance.</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <em>String</em> that represents the current <em>Object</em>. (Inherited from <em>Object</em>.)</td>
</tr>
<tr>
<td>Write</td>
<td>Writes the model to the specified filestream.</td>
</tr>
</tbody>
</table>
See Also

Model Class
Optimization Namespace
Returns an Expression object representing the absolute value of the specified expression.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public static Expression Abs(
    Expression expression,
    double bigM
)
```

Visual Basic

```vbnet
Public Shared Function Abs (_
    expression As Expression, _
    bigM As Double _
) As Expression
```

Visual C++

```cpp
public:
static Expression^ Abs(
    Expression^ expression,
    double bigM
)
```

Parameters

- **expression**
  Type: `Optimization.Expression`
  The expression.

- **bigM**
  Type: `System.Double`
  Large number, only used if the bounds of the variables of the expressions are non-finite
**Return Value**

See Also

Model Class
Optimization Namespace
Model.AddAlternativeConstraintGroups Method


Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public void AddAlternativeConstraintGroups(
    IEnumerable<Constraint> group1,
    IEnumerable<Constraint> group2,
    double bigM
)
```

### Visual Basic

```vbnet
Public Sub AddAlternativeConstraintGroups (_
    group1 As IEnumerable(Of Constraint), _
    group2 As IEnumerable(Of Constraint), _
    bigM As Double _
)
```

### Visual C++

```cpp
public:
void AddAlternativeConstraintGroups(
    IEnumerable<Constraint>^ group1,
    IEnumerable<Constraint>^ group2,
    double bigM
)
```

## Parameters

**group1**
Type: `System.Collections.Generic.IEnumerable<Constraint>`


**group2**
Type: `System.Collections.Generic.IEnumerable<Constraint>`

bigM
Type: System.Double

See Also

Model Class
Optimization Namespace

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public void AddAlternativeConstraints(
    IEnumerable<Constraint> constraints,
    double bigM
)
```

### Visual Basic

```vbnet
Public Sub AddAlternativeConstraints ( _
    constraints As IEnumerable(Of Constraint), _
    bigM As Double _
)
```

### Visual C++

```cpp
public:
void AddAlternativeConstraints(
    IEnumerable<Constraint^>^ constraints,
    double bigM
)
```

## Parameters

- **constraints**
  Type: `System.Collections.Generic.IEnumerable<Optimization.Constraint>`

- **bigM**
  Type: `System.Double`
See Also

Model Class
Optimization Namespace
Optimization Framework

Model.AddConstraint Method

Model Class  See Also  Send Feedback
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>AddConstraint(Constraint, String)</code></td>
<td>Adds <code>constraint</code> to this model instance if it is consistent. If you choose to provide a name, the name of the constraint will be overridden. You need to make sure that the name of the constraint is unique.</td>
</tr>
<tr>
<td><code>AddConstraint(String, Double, Double, Expression)</code></td>
<td>Adds a new constraint to this model instance if it is consistent.</td>
</tr>
</tbody>
</table>
See Also

Model Class
Model Members
Optimization Namespace
Model.AddConstraint Method (Constraint, String)

Adds a constraint to this model instance if it is consistent. If you choose to provide a name, the name of the constraint will be overridden. You need to make sure that the name of the constraint is unique.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
class MyClass
{
    public void AddConstraint(Constraint constraint, string name)
    {
    }
}
```

### Visual Basic

```vbnet
Public Sub AddConstraint(Constraint As Constraint, String As String)
{
}
```

### Visual C++

```cpp
public:
void AddConstraint(Constraint constraint, String name)
{
}
```

### Parameters

- **constraint**
  Type: [Optimization.Constraint](#)
  Constraint to add.

- **name**
  Type: [System.String](#)
  The name for the constraint
See Also

Model Class
AddConstraint Overload
Optimization Namespace
Model.AddConstraint Method (String, Double, Double, Expression)

Adds a new constraint to this model instance if it is consistent.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public void AddConstraint(
    string name,
    double lowerBound,
    double upperBound,
    Expression expression
)
```

**Visual Basic**

```vbnet
Public Sub AddConstraint ( _
    name As String, _
    lowerBound As Double, _
    upperBound As Double, _
    expression As Expression _
)
```

**Visual C++**

```cpp
public:
void AddConstraint(
    String^ name,
    double lowerBound,
    double upperBound,
    Expression^ expression
)
```

### Parameters

**name**

Type: `System.String`

Unique name of the new constraint.
**lowerBound**
Type: `System.Double`
Lower bound (left hand side) of the new constraint.

**upperBound**
Type: `System.Double`
Upper bound (right hand side) of the new constraint.

**expression**
Type: `Optimization.Expression`
Expression of the new constraint.
See Also

Model Class
AddConstraint Overload
Optimization Namespace
Optimization Framework

Model.AddConstraints Method

Adds a set of constraint to this model instance if it is consistent.

Namespace: Optimization

Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public void AddConstraints(
    IEnumerable<Constraint> constraints
)
```

**Visual Basic**

```vbnet
Public Sub AddConstraints (_
    constraints As IEnumerable(Of Constraint) _
)
```

**Visual C++**

```cpp
public:
void AddConstraints(
    IEnumerable<Constraint>^ constraints
)
```

### Parameters

`constraints`  
Type: `System.Collections.Generic.IEnumerable<Constraint>`  
Set of constraints to add.
See Also

Model Class
Optimization Namespace
Model.AddObjective Method

Model Class  See Also  Send Feedback
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>AddObjective(Objective)</code></td>
<td>Adds <em>objective</em> to this model instance if it is consistent.</td>
</tr>
<tr>
<td><code>AddObjective(Expression, String, ObjectiveSense)</code></td>
<td>Adds a new objective to this model instance if it is consistent.</td>
</tr>
</tbody>
</table>
See Also

Model Class
Model Members
Optimization Namespace
Adds *objective* to this model instance if it is consistent.

**Namespace:** [Optimization](https://www.optimization.com)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

public void AddObjective(
    Objective objective
)

Visual Basic

Public Sub AddObjective ( _
    objective As Objective _
)

Visual C++

public:
void AddObjective(
    Objective^ objective
)

Parameters

objective
Type: Optimization.Objective
Objective to add.
See Also

Model Class
AddObjective Overload
Optimization Namespace
Adds a new objective to this model instance if it is consistent.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public void AddObjective(
   Expression expression,
   string name,
   ObjectiveSense sense
)
```

Visual Basic

```vbnet
Public Sub AddObjective (_
   expression As Expression, _
   name As String, _
   sense As ObjectiveSense _
)
```

Visual C++

```cpp
public:
void AddObjective(
   Expression^ expression,
   String^ name,
   ObjectiveSense sense
)
```

Parameters

- **expression**
  Type: **Optimization.Expression**
  Expression of the new objective.

- **name**
  Type: **System.String**
  Unique name of the new objective.
Sense
Type: Optimization, Objective Sense
Sense of the new objective.
See Also

- Model Class
- AddObjective Overload
- Optimization Namespace
Model.AddObjectives Method

Adds a set of objectives to this model instance if it is consistent.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public void AddObjectives(
    IEnumerable<Objective> objectives
)
```

**Visual Basic**

```vbnet
Public Sub AddObjectives (_
    objectives As IEnumerable(Of Objective) _
)
```

**Visual C++**

```cpp
public:
void AddObjectives(
    IEnumerable<Objective>^ objectives
)
```

### Parameters

- **objectives**
  
  Type: `System.Collections.Generic.IEnumerable<Objective>`
  
  Set of objectives to add.
See Also

- Model Class
- Optimization Namespace
Model.AddSOS Method

Adds an SOS Constraint of the given SOSType

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public void AddSOS(
    IDictionary<IVariable, double> variables,
    SOSType type
)
```

**Visual Basic**

```vbnet
Public Sub AddSOS ( _
    variables As IDictionary(Of IVariable, Double)
    type As SOSType _
)
```

**Visual C++**

```cpp
public:
void AddSOS(
    IDictionary<IVariable^, double>^ variables,
    SOSType type
)
```

### Parameters

**variables**
Type: `System.Collections.Generic.IDictionary<IVariable, Double>`
The variables.

**type**
Type: `Optimization,SOSType`
The type.
See Also

Model Class
Optimization Namespace
Optimization Framework

Model.AddSOS1 Method

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0

Adds an SOS1 constraint
Syntax

C#

```csharp
public void AddSOS1(IDictionary<IVariable, double> variables)
```

Visual Basic

```vbnet
Public Sub AddSOS1 (_
    variables As IDictionary(Of IVariable, Double)
)
```

Visual C++

```cpp
public:
    void AddSOS1(I
        IDictionary<IVariable^, double>^ variables
    )
```

Parameters

*variables*
Type: `System.Collections.Generic.IDictionary<IVariable, Double>`
The variables.
See Also

Model Class
Optimization Namespace
Model.AddSOS2 Method

Adds an SOS2 constraint

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public void AddSOS2(
    IDictionary<IVariable, double> variables
)
```

Visual Basic

```vbnet
Public Sub AddSOS2 ( _
    variables As IDictionary(Of IVariable, Double)
)
```

Visual C++

```cpp
public:
void AddSOS2(
    IDictionary<IVariable^, double>^ variables
)
```

Parameters

variables
Type: `System.Collections.Generic.IDictionary<IVariable, Double>
The variables.`
See Also

- Model Class
- Optimization Namespace
Model.AddSOS3 Method

Adds an SOS3 constraint

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public void AddSOS3(
    IDictionary<IVariable, double> variables
)
```

**Visual Basic**

```vbnet
Public Sub AddSOS3 ( 
    variables As IDictionary(Of IVariable, Double) 
)
```

**Visual C++**

```cpp
public:
void AddSOS3(
    IDictionary<IVariable^, double>^ variables
)
```

### Parameters

*variables*

Type: System.Collections.Generic.IDictionary(IVariable, Double)

The variables.
See Also

Model Class
Optimization Namespace
Optimization Framework

Model.AddVariable Method

Model Class  See Also  Send Feedback
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>AddVariable(IVariable)</code></td>
<td>Adds <code>variable</code> to this model instance if it is consistent.</td>
</tr>
<tr>
<td><code>AddVariable(String, Double, Double, VariableType)</code></td>
<td>Adds a new variable to this model instance if it is consistent.</td>
</tr>
</tbody>
</table>
See Also

Model Class
Model Members
Optimization Namespace
Optimization Framework

Model.AddVariable Method (IVariable)

Model Class  See Also  Send Feedback

Adds *variable* to this model instance if it is consistent.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public void AddVariable(
    IVariable variable
)
```

Visual Basic

```vbnet
Public Sub AddVariable (_
    variable As IVariable _
)
```

Visual C++

```cpp
public:
void AddVariable(
    IVariable^ variable
)
```

Parameters

variable
Type: Optimization.Interfaces.IVariable
Variable to add.
See Also

Model Class
AddVariable Overload
Optimization Namespace
Model.AddVariable Method (String, Double, Double, VariableType)

Adds a new variable to this model instance if it is consistent.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public void AddVariable(
    string name,
    double lowerBound,
    double upperBound,
    VariableType type
)
```

Visual Basic

```vbnet
Public Sub AddVariable ( _
    name As String, _
    lowerBound As Double, _
    upperBound As Double, _
    type As VariableType _
)
```

Visual C++

```cpp
public:
    void AddVariable(
        String^ name,
        double lowerBound,
        double upperBound,
        VariableType type
    )
```

Parameters

name
Type: System.String
Unique name of the new variable.
lowerBound
Type: System.Double
Lower bound of the new variable.

upperBound
Type: System.Double
Upper bound of the new variable.

type
Type: Optimization.VariableType
Type of the new variable.
See Also

Model Class
AddVariable Overload
Optimization Namespace
Model.AddVariables Method

Adds a set of variables to this model instance if it is consistent. //TODO: discuss: What do we mean by "consistent"?

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

**C#**

```csharp
public void AddVariables(
    IEnumerable<IVariable> variables
)
```

**Visual Basic**

```vbnet
Public Sub AddVariables ( _
    variables As IEnumerable(Of IVariable) _
)
```

**Visual C++**

```cpp
public:
void AddVariables(
    IEnumerable<IVariable>^ variables
)
```

**Parameters**

*variables*

Type: **System.Collections.Generic.IEnumerable<IVariable>**

Set of variables to add.
See Also

Model Class
Optimization Namespace
Returns an Expression object representing the ceil roundoff of the specified expression.

**Namespace:** [Optimization](https://www.optimization.com)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td><code>public static Expression Ceil(Expression expression)</code></td>
</tr>
<tr>
<td>Visual Basic</td>
<td><code>Public Shared Function Ceil (_expression As Expression _) As Expression</code></td>
</tr>
</tbody>
</table>
| Visual C++ | `public:
static Expression^ Ceil(Expression^ expression)`               |

### Parameters

- **expression**
  - Type: `Optimization.Expression`
  - The expression.

### Return Value

See Also

Model Class
Optimization Namespace
Removes all variables, constraints, objectives and scenarios from this model instance.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th></th>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>public void Clear()</td>
<td>Public Sub Clear</td>
<td>public: void Clear()</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
See Also

- Model Class
- Optimization Namespace
Contains this model instance a constraint named *name*?

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public bool ContainsConstraint(
    string name
)
```

### Visual Basic

```vbnet
Public Function ContainsConstraint ( _
    name As String _
) As Boolean
```

### Visual C++

```cpp
public:
    bool ContainsConstraint(
        String^ name
    )
```

## Parameters

- **name**
  - Type: `System.String`
  - Name of the constraint to search for.

## Return Value

See Also

Model Class
Optimization Namespace
Contains this model instance an objective named *name*?

**Namespace:** [Optimization](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
Syntax

C#

```csharp
public bool ContainsObjective(
    string name
)
```

Visual Basic

```vbnet
Public Function ContainsObjective ( _
    name As String _
) As Boolean
```

Visual C++

```cpp
public:
bool ContainsObjective(
    String^ name
)
```

Parameters

`name`
Type: `System.String`
Name of the objective to search for.

Return Value

See Also

Model Class
Optimization Namespace
Contains this model instance a variable named *name*?

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
## Syntax

### C#

```csharp
public bool ContainsVariable(
    string name
)
```

### Visual Basic

```vbnet
Public Function ContainsVariable ( _
    name As String _
) As Boolean
```

### Visual C++

```cpp
public:
bool ContainsVariable(
    String^ name
)
```

## Parameters

- **name**
  - Type: `System.String`
  - Name of the variable to search for.

## Return Value

See Also

Model Class
Optimization Namespace
Optimization Framework

Model.Floor Method

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0

Returns an Expression object representing the floor roundoff of the specified expression.
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>public static Expression Floor( Expression expression )</td>
<td>Public Shared Function Floor ( _ expression As Expression _) As Expression</td>
<td>public: static Expression^ Floor( Expression^ expression )</td>
</tr>
</tbody>
</table>

### Parameters

*expression*

Type: [Optimization.Expression](#)

The expression.

### Return Value

See Also

Model Class
Optimization Namespace
Model.GetConstraint Method

Returns the constraint named *name*, or *null* if this model instance contains no such constraint.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#**   | public `Constraint` GetConstraint(
|          |     `string` name
|          | ) |
| **Visual Basic** | Public Function GetConstraint ( _
|                  |    `name` As `String` _
|                  | ) As `Constraint` |
| **Visual C++**  | public:
|                | `Constraint`^ GetConstraint(
|                |    `String`^ name
|                | ) |

### Parameters

**name**

Type: `System.String`

Name of the constraint to search for.

### Return Value

See Also

- Model Class
- Optimization Namespace
Returns the objective named *name*, or *null* if this model instance contains no such objective.

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

**C#**

```csharp
public Objective GetObjective(
    string name
)
```

**Visual Basic**

```vbnet
Public Function GetObjective ( _
    name As String _
) As Objective
```

**Visual C++**

```cpp
public:
Objective^ GetObjective(
    String^ name
)
```

### Parameters

**name**

Type: `System.String`

Name of the objective to search for.

### Return Value

See Also

Model Class
Optimization Namespace
Model.GetVariable Method

Returns the variable named name, or null if this model instance contains no such variable.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public IVariable GetVariable(
    string name
)
```

**Visual Basic**

```vbnet
Public Function GetVariable ( _
    name As String _
) As IVariable
```

**Visual C++**

```cpp
public:
IVariable^ GetVariable(
    String^ name
)
```

### Parameters

**name**

Type: `System.String`

Name of the variable to search for.

### Return Value

See Also

Model Class
Optimization Namespace
Model.Load Method

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0

Loads the specified file.
## Syntax

### C#

```csharp
public void Load(
    Stream filestream
)
```

### Visual Basic

```vbnet
Public Sub Load (_
    filestream As Stream _
)
```

### Visual C++

```cpp
public:
void Load(  
    Stream^ filestream
)
```

## Parameters

**filestream**

Type: `System.IO.Stream`

The filestream.
See Also

Model Class
Optimization Namespace
Model.Max Method

Returns an Expression object representing the maximum of the specified expressions.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>public static Expression Max( IEnumerable&lt;Expression&gt; expressions, double bigM )</td>
</tr>
<tr>
<td>Visual Basic</td>
<td>Public Shared Function Max ( _ expressions As IEnumerable(Of Expression), _ bigM As Double _ ) As Expression</td>
</tr>
<tr>
<td>Visual C++</td>
<td>public: Expression^ Max( IEnumerable&lt;Expression^&gt;^ expressions, double bigM )</td>
</tr>
</tbody>
</table>

## Parameters

- **expressions**
  - Type: System.Collections.Generic.IEnumerable<Expression>
  - The expressions.

- **bigM**
  - Type: System.Double
  - Large number, only used if the bounds of the variables of the expressions are non-finite
Return Value

[Missing <returns> documentation for
See Also

Model Class
Optimization Namespace
Returns an Expression object representing the minimum of the specified expressions.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public static Expression Min(
    IEnumerable<Expression> expressions,
    double bigM
)
```

### Visual Basic

```vbnet
Public Shared Function Min ( _
    expressions As IEnumerable(Of Expression), _
    bigM As Double _
) As Expression
```

### Visual C++

```cpp
public: Expression^ Min(
    IEnumerable<Expression^>^ expressions,
    double bigM
)
```

## Parameters

- **expressions**
  - Type: `System.Collections.Generic.IEnumerable<Expression>`
  - The expressions.

- **bigM**
  - Type: `System.Double`
  - Large number, only used if the bounds of the variables of the expressions are non-finite
Return Value

See Also

Model Class
Optimization Namespace
Removes the constraint named *name* from this model instance, or does nothing if this model instance contains no such constraint.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public void RemoveConstraint(
    string name
)
```

Visual Basic

```vbnet
Public Sub RemoveConstraint ( _
    name As String _
)
```

Visual C++

```cpp
public:
    void RemoveConstraint(
        String^ name
    )
```

Parameters

name
Type: System.String
Name of the constraint to remove.
See Also

Model Class
Optimization Namespace
Removes a set of constraints from this model instance.

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#**   | ```
public void RemoveConstraints(
    IEnumerable<string> names
)``` |
| **Visual Basic** | ```
Public Sub RemoveConstraints (_
    names As IEnumerable(Of String) _
)``` |
| **Visual C++** | ```
public:
void RemoveConstraints(
    IEnumerable<String[^>] names
)``` |

### Parameters

**names**
Type: `System.Collections.Generic.IEnumerable<String>
Names of the constraints to remove.`
See Also

Model Class
Optimization Namespace
Removes the objective named *name* from this model instance, or does nothing if this model instance contains no such objective.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public void RemoveObjective(</td>
</tr>
<tr>
<td></td>
<td>string name</td>
</tr>
<tr>
<td></td>
<td>)</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Sub RemoveObjective ( _</td>
</tr>
<tr>
<td></td>
<td>name As String _</td>
</tr>
<tr>
<td></td>
<td>)</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public:</td>
</tr>
<tr>
<td></td>
<td>void RemoveObjective(</td>
</tr>
<tr>
<td></td>
<td>String^ name</td>
</tr>
<tr>
<td></td>
<td>)</td>
</tr>
</tbody>
</table>

### Parameters

- **name**
  - Type: System.String
  - Name of the objective to remove.
See Also

Model Class
Optimization Namespace
Removes a set of objectives from this model instance.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| C#       | ```
    public void RemoveObjectives(
        IEnumerable<string> names
    )
``` |
| Visual Basic | ```
    Public Sub RemoveObjectives ( _
        names As IEnumerable(Of String) _
    )
``` |
| Visual C++ | ```
    public:
    void RemoveObjectives(
        IEnumerable<String^>^ names
    )
``` |

### Parameters

`names`

Type: `System.Collections.Generic.IEnumerable<String>

Names of the objectives to remove.
See Also

Model Class
Optimization Namespace
Removes the variable named *name* from this model instance, or does nothing if this model instance contains no such variable.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

**C#**

```csharp
public void RemoveVariable(
    string name
)
```

**Visual Basic**

```vbnet
Public Sub RemoveVariable (_
    name As String _
)
```

**Visual C++**

```cpp
public:
void RemoveVariable(
    String^ name
)
```

**Parameters**

- **name**
  - Type: `System.String`
  - Name of the variable to remove.
See Also

Model Class
Optimization Namespace
RemoveVariables Method

Removes a set of variables from this model instance.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public void RemoveVariables(
    IEnumerable<string> names
)
```

**Visual Basic**

```vbnet
Public Sub RemoveVariables (_
    names As IEnumerable(Of String) _
)
```

**Visual C++**

```cpp
public:
void RemoveVariables(    
    IEnumerable<String^>^ names
)
```

### Parameters

`names`
Type: `System.Collections.Generic.IEnumerable<String>`
Names of the variables to remove.
See Also

Model Class
Optimization Namespace
Model.Write Method

Writes the model to the specified filestream.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public void Write(
    Stream filestream,
    FileType fileType
)
```

### Visual Basic

```vbnet
Public Sub Write ( _
    filestream As Stream, _
    fileType As FileType _
)
```

### Visual C++

```cpp
public:
void Write(
    Stream^ filestream,
    FileType fileType
)
```

## Parameters

- **filestream**
  Type: `System.IO.Stream`
  The filestream.

- **fileType**
  Type: `Optimization.FileType`
  Type of the file.
See Also

Model Class
Optimization Namespace
The **Model** type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AreAllConstraintsLinear</td>
<td>Have all constraints in this model instance linear expressions?</td>
</tr>
<tr>
<td>AreAllObjectivesLinear</td>
<td>Have all objectives in this model instance linear expressions?</td>
</tr>
<tr>
<td>AreAllVariablesBinary</td>
<td>Are all variables in this model instance of type integer and their lower bound equal to zero and their upper bound equal to 1?</td>
</tr>
<tr>
<td>AreAllVariablesContinuous</td>
<td>Are variables in this model instance of type continuous?</td>
</tr>
<tr>
<td>AreAllVariablesInteger</td>
<td>Are all variables in this model instance of type integer?</td>
</tr>
<tr>
<td>Constraints</td>
<td>Constraints in this model instance.</td>
</tr>
<tr>
<td>ConstraintsCount</td>
<td>Count of constraints in this model instance.</td>
</tr>
<tr>
<td>ModelBehavior</td>
<td>Gets or sets the behavior of the model. Available options are &quot;Automatic&quot; and &quot;Manual&quot; In automatic mode, the model takes care of adding and removing variables when constraints and objectives get added. In manual mode you need to take care of this yourself.</td>
</tr>
<tr>
<td>Name</td>
<td>Name of this model instance.</td>
</tr>
<tr>
<td>Objectives</td>
<td>Objectives in this model instance.</td>
</tr>
<tr>
<td>ObjectivesCount</td>
<td>Count of objectives in this model instance.</td>
</tr>
<tr>
<td>SOS1Sets</td>
<td>Sets of SOS1 variables in this model instance.</td>
</tr>
<tr>
<td>SOS2Sets</td>
<td>Sets of SOS2 variables in this model instance.</td>
</tr>
<tr>
<td>SOS3Sets</td>
<td>Sets of SOS3 variables in this model instance.</td>
</tr>
<tr>
<td>Variables</td>
<td>Variables in this model instance.</td>
</tr>
<tr>
<td>-----------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>VariablesCount</td>
<td>Count of variables in this model instance.</td>
</tr>
</tbody>
</table>
See Also

Model Class
Optimization Namespace
Have all constraints in this model instance linear expressions?

**Namespace:** [Optimization](https://www.optimization.com)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public bool AreAllConstraintsLinear { get; }
```

Visual Basic

```vbnet
Public ReadOnly Property AreAllConstraintsLinear As Boolean
    Get
```

Visual C++

```cpp
public:
    property bool AreAllConstraintsLinear {
        bool get ();
    }
```

Field Value
See Also

Model Class
Optimization Namespace
Have all objectives in this model instance linear expressions?

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public bool AreAllObjectivesLinear { get; }
```

### Visual Basic

```vbnet
Public ReadOnly Property AreAllObjectivesLinear As Boolean
Get
```

### Visual C++

```cpp
public:
property bool AreAllObjectivesLinear {
    bool get ();
}
```

## Field Value
See Also

Model Class
Optimization Namespace
Are all variables in this model instance of type integer and their lower bound equal to zero and their upper bound equal to 1?

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public bool AreAllVariablesBinary { get; }
```

**Visual Basic**

```vbscript
Public ReadOnly Property AreAllVariablesBinary As Boolean
    Get

Field Value
See Also

Model Class
Optimization Namespace
Are variables in this model instance of type continuous?

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public bool AreAllVariablesContinuous { get; }
```

### Visual Basic

```vbnet
Public ReadOnly Property AreAllVariablesContinuous As Get
```

### Visual C++

```cpp
public:
property bool AreAllVariablesContinuous {
    bool get ();
}
```

## Field Value
See Also

Model Class
Optimization Namespace
Are all variables in this model instance of type integer?

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public bool AreAllVariablesInteger { get; }
```

**Visual Basic**

```vbnet
Public ReadOnly Property AreAllVariablesInteger As Boolean
     Get
```

**Visual C++**

```cpp
public:
    property bool AreAllVariablesInteger {
        bool get ();
    }
```

### Field Value
See Also

Model Class
Optimization Namespace
Constraints in this model instance.

**Namespace:** [Optimization](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public IEnumerable&lt;Constraint&gt; Constraints { get; }</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public ReadOnly Property Constraints As IEnumerable(Of Constraint) Get</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public: property IEnumerable&lt;Constraint&gt;^ Constraints { IEnumerable&lt;Constraint&gt;^ get (); }</code></td>
</tr>
</tbody>
</table>

### Field Value
See Also

- Model Class
- Optimization Namespace
Count of constraints in this model instance.

**Namespace:** [Optimization](https://www.example.com)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public int ConstraintsCount { get; }
```

### Visual Basic

```vbnet
Public ReadOnly Property ConstraintsCount As Integer
    Get
```

### Visual C++

```cpp
public:
    property int ConstraintsCount {
        int get ();
    }
```

## Field Value
See Also

Model Class
Optimization Namespace
Gets or sets the behavior of the model. Available options are "Automatic" and "Manual" In automatic mode, the model takes care of adding and removing variables when constraints and objectives get added. In manual mode you need to take care of this yourself.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
</table>
| public `ModelBehavior` ModelBehavior { get; set; } | Public Property ModelBehavior As `ModelBehavior`  
Get  
Set | public: property `ModelBehavior` ModelBehavior {  
`ModelBehavior` get ();  
void set (`ModelBehavior` value); |

**Field Value**

The model behavior.
See Also

- Model Class
- Optimization Namespace
Optimization Framework

Model.Name Property

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0

Name of this model instance.
### Syntax

**C#**

```csharp
public string Name { get; set; }
```

**Visual Basic**

```vbnet
Public Property Name As String
    Get
    Set
```

**Visual C++**

```cpp
public:
    property String^ Name {
        String^ get ();
        void set (String^ value);
    }
```

### Field Value
See Also

Model Class
Optimization Namespace
Objectives in this model instance.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public IEnumerable<Objective> Objectives { get; }
```

**Visual Basic**

```vbnet
Public ReadOnly Property Objectives As IEnumerable(Of Objective)
Get
```

**Visual C++**

```cpp
public:
property IEnumerable<Objective^>^ Objectives {
    IEnumerable<Objective^>^ get ();
}
```

### Field Value
See Also

Model Class
Optimization Namespace
Count of objectives in this model instance.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public int ObjectivesCount { get; }
```

#### Visual Basic

```vbnet
Public ReadOnly Property ObjectivesCount As Integer
    Get
End Property
```

#### Visual C++

```cpp
public:
    property int ObjectivesCount
        { int get (); }
```

### Field Value
See Also

- Model Class
- Optimization Namespace
Model.SOS1Sets Property

Sets of SOS1 variables in this model instance.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public IEnumerable&lt;IDictionary&lt;IVariable, double&gt;&gt; SOS1Sets</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public ReadOnly Property SOS1Sets As IEnumerable(Of IDictionary&lt;IVariable, double&gt;)</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: property IEnumerable&lt;IDictionary&lt;IVariable^, double&gt;^&gt;^ IEnumerable&lt;IDictionary&lt;IVariable^, double&gt;^&gt;^</td>
</tr>
</tbody>
</table>

### Field Value
See Also

Model Class
Optimization Namespace
Sets of SOS2 variables in this model instance.

**Namespace:** [Optimization](https://example.com)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td><code>public IEnumerable&lt;IDictionary&lt;IVariable, double&gt;&gt; SOS2Sets</code></td>
</tr>
<tr>
<td>Visual Basic</td>
<td><code>Public ReadOnly Property SOS2Sets As IEnumerable(Of IEnumerable&lt;IDictionary&lt;IVariable, double&gt;&gt;)</code></td>
</tr>
<tr>
<td>Visual C++</td>
<td><code>public: property IEnumerable&lt;IDictionary&lt;IVariable^, double&gt;^, IEnumerable&lt;IDictionary&lt;IVariable^, double&gt;^&gt;</code></td>
</tr>
</tbody>
</table>

### Field Value
See Also

Model Class
Optimization Namespace
Sets of SOS3 variables in this model instance.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public IEnumerable<IDictionary<IVariable, double>> SOS3Sets
```

#### Visual Basic

```vbnet
Public ReadOnly Property SOS3Sets As IEnumerable(Of IDictionary<IVariable, Double>)
```

#### Visual C++

```cpp
public:
    property IEnumerable<IDictionary<IVariable^, double>^>^ SOS3Sets
```

### Field Value
See Also

Model Class
Optimization Namespace
Variables in this model instance.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public <code>IEnumerable&lt;IVariable&gt;</code> Variables { get; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public ReadOnly Property Variables As <code>IEnumerable(Of Get</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: <code>IEnumerable&lt;IVariable&gt;</code> Variables { get (); }</td>
</tr>
</tbody>
</table>

**Field Value**
See Also

Model Class
Optimization Namespace
Model.VariablesCount Property

Count of variables in this model instance.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public int VariablesCount { get; }</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public ReadOnly Property VariablesCount As Integer Get</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public: property int VariablesCount { int get (); }</code></td>
</tr>
</tbody>
</table>
See Also

Model Class
Optimization Namespace
Describes the behavior of a model when adding constraints

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
**Syntax**

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public enum ModelBehavior</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Enumeration ModelBehavior</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public enum class ModelBehavior</code></td>
</tr>
</tbody>
</table>
## Members

<table>
<thead>
<tr>
<th>Member name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manual</td>
<td>0</td>
<td>Variables in a constraint will not be added to the model, you need to do this manually beforehand</td>
</tr>
<tr>
<td>Auto</td>
<td>1</td>
<td>Variables in a constraint will be added to the model automatically</td>
</tr>
</tbody>
</table>
See Also

Optimization Namespace
Represent an objective

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>[SerializableAttribute]</code></td>
</tr>
<tr>
<td><code>public class Objective</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual Basic</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;SerializableAttribute&gt;</code></td>
</tr>
<tr>
<td><code>Public Class Objective</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>[SerializableAttribute]</code></td>
</tr>
<tr>
<td><code>public ref class Objective</code></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
Optimization.Objective
See Also

Objective Members
Optimization Namespace
The **Objective** type exposes the following members.
Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objective</td>
<td>Initializes a new instance of the Objective class.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is equal to the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.type">Type</a> of the current instance. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="https://docs.microsoft.com/en-us/dotnet/api/system.string">String</a> that represents the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expression</td>
<td>Expression of this objective.</td>
</tr>
<tr>
<td>Name</td>
<td>Unique name of this objective.</td>
</tr>
<tr>
<td>Sense</td>
<td>Sense of this objective.</td>
</tr>
</tbody>
</table>
See Also

Objective Class
Optimization Namespace
Initializes a new instance of the **Objective** class.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public Objective(
    Expression expression,
    string name,
    ObjectiveSense sense
)
```

### Visual Basic

```vbnet
Public Sub New ( _
    expression As Expression, _
    name As String, _
    sense As ObjectiveSense _
)
```

### Visual C++

```cpp
public:
Objective(  
    Expression expression,
    String name,
    ObjectiveSense sense
)
```

## Parameters

**expression**
Type: `Optimization.Expression`

The expression.

**name**
Type: `System.String`

The name.
The sense.

Type: Optimization, Objective Sense
See Also

Objective Class
Optimization Namespace
The **Objective** type exposes the following members.
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <em>Object</em> is equal to the current <em>Object</em>. (Inherited from <em>Object</em>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <em>Object</em> to attempt to free resources and perform other cleanup operations before the <em>Object</em> is reclaimed by garbage collection. (Inherited from <em>Object</em>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <em>Object</em>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <em>Type</em> of the current instance. (Inherited from <em>Object</em>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <em>Object</em>. (Inherited from <em>Object</em>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <em>String</em> that represents the current <em>Object</em>. (Inherited from <em>Object</em>.)</td>
</tr>
</tbody>
</table>
See Also

Objective Class
Optimization Namespace
The **Objective** type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expression</td>
<td>Expression of this objective.</td>
</tr>
<tr>
<td>Name</td>
<td>Unique name of this objective.</td>
</tr>
<tr>
<td>Sense</td>
<td>Sense of this objective.</td>
</tr>
</tbody>
</table>
See Also

Objective Class
Optimization Namespace
Expression of this objective.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>public Expression Expression { get; set; }</td>
</tr>
<tr>
<td>Visual Basic</td>
<td>Public Property Expression As Expression Get Set</td>
</tr>
<tr>
<td>Visual C++</td>
<td>public: property Expression^ Expression { Expression^ get (); void set (Expression^ value); }</td>
</tr>
</tbody>
</table>

### Field Value
See Also

Objective Class
Optimization Namespace
Unique name of this objective.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public string Name { get; set; }
```

#### Visual Basic

```vbnet
Public Property Name As String
    Get
    Set
```

#### Visual C++

```cpp
public:
    property String^ Name {
        String^ get ();
        void set (String^ value);
    }
```

### Field Value
See Also

Objective Class
Optimization Namespace
Sense of this objective.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public ObjectiveSense Sense { get; set; }
```

#### Visual Basic

```vbnet
Public Property Sense As ObjectiveSense
    Get
    Set
```

#### Visual C++

```cpp
public:
    property ObjectiveSense Sense {
        ObjectiveSense get ();
        void set (ObjectiveSense value);
    }
```

### Field Value
See Also

Objective Class
Optimization Namespace
Sense of an ![Objective](Optimization).
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>public enum ObjectiveSense</code></td>
<td><code>Public Enumeration ObjectiveSense</code></td>
<td><code>public enum class ObjectiveSense</code></td>
</tr>
</tbody>
</table>
### Members

<table>
<thead>
<tr>
<th>Member name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimize</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Maximize</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
See Also

Optimization Namespace
A parameterCollection allows you to define parameters on Sets so that you can access parameter values with indices like in other modelling languages

**Namespace**: Optimization

**Assembly**: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
<tr>
<td><code>public class ParameterCollection&lt;T&gt;</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td></td>
</tr>
<tr>
<td><code>Public Class ParameterCollection(Of T)</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
</tr>
<tr>
<td><code>generic&lt;typename T&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>public ref class ParameterCollection</code></td>
<td></td>
</tr>
</tbody>
</table>
Type Parameters

$T$

[Missing <typeparam name="T"/> documentation for "T:Optimization.ParameterCollection`1"]
Inheritance Hierarchy

System.Object

Optimization.ParameterCollection(T)
See Also

ParameterCollection(T) Members
Optimization Namespace
The `ParameterValueCollection(T)` type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ParameterCollection(T)</td>
<td>Initializes a new instance of the ParameterCollection(T) class.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item</td>
<td>Gets the Double with the specified i.</td>
</tr>
</tbody>
</table>
See Also

ParameterCollection(T) Class
Optimization Namespace
ParameterCollection\((T)\) Constructor

Initializes a new instance of the ParameterCollection\((T)\) class.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Parameters

**parameterValues**
Type: `System.Collections.Generic.IEnumerable<T>`
The parameter values.

**valueExpression**
Type: `System.Func<T, Double>`
The value expression.
sets
Type: Optimization.Interfaces.ISetAccessor(T, Object)[]
The sets.
# Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>parameterValues != null</code></td>
</tr>
</tbody>
</table>

*Description:* You need to provide a `parameterValues` collection.

[Learn more about contracts](#)
See Also

ParameterCollection(T) Class
Optimization Namespace
The **ParameterCollection(T)** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified Object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a String that represents the current Object. (Inherited from Object.)</td>
</tr>
</tbody>
</table>
See Also

ParameterCollection(T) Class
Optimization Namespace
The `ParameterCollection<T>` type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item</td>
<td>Gets the <strong>Double</strong> with the specified i.</td>
</tr>
</tbody>
</table>
See Also

ParameterCollection(T) Class
Optimization Namespace
ParameterCollection(T).Item Property

ParameterCollection(T) Class  See Also  Send Feedback

Gets the Double with the specified i.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Package</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td><code>public double this[params Object[] i] { get; }</code></td>
</tr>
<tr>
<td>Visual Basic</td>
<td><code>Public ReadOnly Default Property Item ( _ParamArray i As Object() _ ) As Double Get</code></td>
</tr>
<tr>
<td>Visual C++</td>
<td><code>public: double default[... array&lt;Object[^]&gt;^ i] { double get (... array&lt;Object[^]&gt;^ i); }</code></td>
</tr>
</tbody>
</table>

### Parameters

- **i**
  - Type: `System.Object[]`

### Field Value
See Also

ParameterCollection(T) Class
Optimization Namespace
Represents a solution of an Model.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th><strong>C#</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>public class Solution : IComparable&lt;Solution&gt;, IEquatable&lt;Solution&gt;, IEquatable&lt;IDictionary&gt;</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Visual Basic</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Public Class Solution _ Implements IComparable(Of Solution), IEquatable(Of IDictionary(Of String, Double))</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Visual C++</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>public ref class Solution : IComparable&lt;Solution^&gt;, IEquatable&lt;Solution^&gt;, IEquatable&lt;IDictionary&gt;</code></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
Optimization.Solution
See Also

Solution Members
Optimization Namespace
The **Solution** type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Solution</code></td>
<td>Creates a new solution instance.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ClearValues</strong></td>
<td>Removes the variable values and objective values from this solution instance and sets the status to <strong>NoSolutionValues</strong>.</td>
</tr>
<tr>
<td><strong>CompareTo</strong></td>
<td>Compares the objective values of this solution instance to others.</td>
</tr>
<tr>
<td><strong>Equals(Object)</strong></td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>Equals(IDictionary(String, Double))</strong></td>
<td>Indicates whether the variable values of this solution instance are equal to otherVariableValues.</td>
</tr>
<tr>
<td><strong>Equals(Solution)</strong></td>
<td>Indicates whether the objective values of this solution instance are equal to others.</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>GetObjectiveValue</strong></td>
<td>Returns the solution value of the objective named name, or null if the status of this solution instance is <strong>NoSolutionValues</strong>.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <strong>Type</strong> of the current instance. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>GetVariableValue</strong></td>
<td>Returns the solution value of the variable named name, or null if the status of this solution instance is <strong>NoSolutionValues</strong>.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ConflictingSet</strong></td>
<td>Gets the conflicting set.</td>
</tr>
<tr>
<td><strong>DualVariableValues</strong></td>
<td>Unique name and solution value for each dual variable in the solved model instance, or \texttt{null} if the status of this solution instance is \texttt{NoSolutionValues}.</td>
</tr>
<tr>
<td><strong>ModelName</strong></td>
<td>Gets the name of the model.</td>
</tr>
<tr>
<td><strong>ModelStatus</strong></td>
<td>Gets the model status.</td>
</tr>
<tr>
<td><strong>ObjectiveValues</strong></td>
<td>Unique name and solution value for each objective in the solved model instance, or \texttt{null} if the status of this solution instance is \texttt{NoSolutionValues}.</td>
</tr>
<tr>
<td><strong>OverallWallTime</strong></td>
<td>Gets the overall wall time.</td>
</tr>
<tr>
<td><strong>Status</strong></td>
<td>Status of this solution instance.</td>
</tr>
<tr>
<td><strong>VariableValues</strong></td>
<td>Unique name and solution value for each variable in the solved model instance, or \texttt{null} if the status of this solution instance is \texttt{NoSolutionValues}.</td>
</tr>
</tbody>
</table>
See Also

Solution Class
Optimization Namespace
Optimization Framework

Solution Constructor

Solution Class  See Also  Send Feedback

Creates a new solution instance.

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
public Solution(
    string modelName,
    TimeSpan overallWallTime,
    ModelStatus modelStatus,
    SolutionStatus status,
    IDictionary<string, double> variableValues,
    IDictionary<string, double> dualVariableValues,
    IDictionary<string, double> objectiveValues,
    ConflictingSet conflictingSet)

public Sub New (_
    modelName As String, _
    overallWallTime As TimeSpan, _
    modelStatus As ModelStatus, _
    status As SolutionStatus, _
    variableValues As IDictionary(Of String, Double),
    dualVariableValues As IDictionary(Of String, Double),
    objectiveValues As IDictionary(Of String, Double),
    conflictingSet As ConflictingSet _
)

public:
    Solution(
        String^ modelName,
        TimeSpan overallWallTime,
        ModelStatus modelStatus,
        SolutionStatus status,
Parameters

modelName
Type: System.String
Name of the solved model instance.

overallWallTime
Type: System.TimeSpan
Overall wall clock time taken for solving.

modelStatus
Type: Optimization.Solver.ModelStatus
Status of the solved model instance.

status
Type: Optimization.Solver.SolutionStatus
Status of the new solution instance.

variableValues
Type: System.Collections.Generic.IDictionary(String, Double)
Name and solution value for each variable in the solved model instance, or null if the status of this solution instance is NoSolutionValues.

dualVariableValues
Type: System.Collections.Generic.IDictionary(String, Double)
Name and solution value for each dual variable in the solved model instance, or null if the status of this solution instance is NoSolutionValues.

objectiveValues
Type: System.Collections.Generic.IDictionary(String, Double)
Name and solution value for each objective in the solved model instance, or null if the status of this solution instance is NoSolutionValues.

conflictingSet
Type: Optimization.ConflictingSet
Missing <param name="conflictingSet"/> documentation for
See Also

Solution Class
Optimization Namespace
The **Solution** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ClearValues</strong></td>
<td>Removes the variable values and objective values from this solution instance and sets the status to NoSolutionValues.</td>
</tr>
<tr>
<td><strong>CompareTo</strong></td>
<td>Compares the objective values of this solution instance to others.</td>
</tr>
<tr>
<td><strong>Equals(Object)</strong></td>
<td>Determines whether the specified Object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>Equals(IDictionary(String, Double))</strong></td>
<td>Indicates whether the variable values of this solution instance are equal to otherVariableValues.</td>
</tr>
<tr>
<td><strong>Equals(Solution)</strong></td>
<td>Indicates whether the objective values of this solution instance are equal to others.</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetObjectiveValue</strong></td>
<td>Returns the solution value of the objective named name, or null if the status of this solution instance is NoSolutionValues.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetVariableValue</strong></td>
<td>Returns the solution value of the variable named name, or null if the status of this solution instance is NoSolutionValues.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
</tbody>
</table>
See Also

Solution Class
Optimization Namespace
Removes the variable values and objective values from this solution instance and sets the status to **NoSolutionValues**.

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
## Syntax

### C#

```csharp
public virtual void ClearValues()
```

### Visual Basic

```vbnet
Public Overridable Sub ClearValues
```

### Visual C++

```cpp
public:
virtual void ClearValues()
```
See Also

Solution Class
Optimization Namespace
Solution.CompareTo Method

Compares the objective values of this solution instance to others.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>public int CompareTo(Solution other)</code></td>
<td><code>Public Function CompareTo (_other As Solution _) As Integer</code></td>
<td><code>public: virtual int CompareTo(Solution^ other)</code> sealed</td>
</tr>
</tbody>
</table>

### Parameters

`other`  
Type: `Optimization.Solution`  
Solution instance to compare to.

### Return Value

If the objective values are compared, the return value indicates dominance of solution quality (i.e. zero indicates indifference).

### Implements

`IComparable<T>.CompareTo<T>`
## Exceptions

<table>
<thead>
<tr>
<th>Exception</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>System.NotSupportedException</td>
<td>If this solution instance has status NoSolutionValues.</td>
</tr>
<tr>
<td>System.ArgumentException</td>
<td>If other has status NoSolutionValues or the solutions' objective names do not fit to each other.</td>
</tr>
</tbody>
</table>
See Also

Solution Class
Optimization Namespace
Optimization Framework

Solution.Equals Method

Solution Class See Also Send Feedback
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals(Object)</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>Equals(IDictionary(String, Double))</strong></td>
<td>Indicates whether the variable values of this solution instance are equal to otherVariableValues.</td>
</tr>
<tr>
<td><strong>Equals(Solution)</strong></td>
<td>Indicates whether the objective values of this solution instance are equal to others.</td>
</tr>
</tbody>
</table>
See Also

Solution Class
Solution Members
Optimization Namespace
Indicates whether the variable values of this solution instance are equal to `otherVariableValues`.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public bool Equals(
    IDictionary<string, double> otherVariableValues)
```

Visual Basic

```vbnet
Public Function Equals (_
    otherVariableValues As IDictionary(Of String,
) As Boolean
```

Visual C++

```cpp
public:
    virtual bool Equals(
        IDictionary<String^, double>^ otherVariableValues)
```

Parameters

`otherVariableValues`
Type: `System.Collections.Generic.IDictionary<String, Double>`
The variable values to compare to.

Return Value

[Missing <returns> documentation for

Implements

IEquatable<T>.Equals(T)"
### Exceptions

<table>
<thead>
<tr>
<th>Exception</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>System.NotSupportedException</code></td>
<td>If this solution instance has status <code>NoSolutionValues</code>.</td>
</tr>
<tr>
<td><code>System.ArgumentException</code></td>
<td>If the solutions' variable names do not fit to each other.</td>
</tr>
</tbody>
</table>
See Also

Solution Class
Equals Overload
Optimization Namespace
Indicates whether the objective values of this solution instance are equal to others.

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td><code>public bool Equals(Solution other)</code></td>
</tr>
<tr>
<td>Visual Basic</td>
<td><code>Public Function Equals(_ other As Solution _) As Boolean</code></td>
</tr>
<tr>
<td>Visual C++</td>
<td><code>public: virtual bool Equals(Solution^ other)</code></td>
</tr>
</tbody>
</table>

### Parameters

- **other**
  - Type: `Optimization.Solution`
  - The `Solution` to compare to.

### Return Value


### Implements

- `IEquatable(T).Equals(T)`
## Exceptions

<table>
<thead>
<tr>
<th>Exception</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>System.InvalidOperationException</code></td>
<td>If this solution instance has status <code>NoSolutionValues</code>.</td>
</tr>
<tr>
<td><code>System.ArgumentException</code></td>
<td>If <em>other</em> has status <code>NoSolutionValues</code> or the solutions' objective names do not fit to each other.</td>
</tr>
</tbody>
</table>
See Also

Solution Class
Equals Overload
Optimization Namespace
Returns the solution value of the objective named *name*, or `null` if the status of this solution instance is `NoSolutionValues`.

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
Syntax

C#

public Nullable<double> GetObjectiveValue(string name)

Visual Basic

Public Function GetObjectiveValue(_) As Nullable(Of Double)
    name As String_

Visual C++

public:
    Nullable<double> GetObjectiveValue(String^ name)

Parameters

name
Type: System.String
Name of the objective.

Return Value

See Also

Solution Class
Optimization Namespace
Returns the solution value of the variable named \textit{name}, or \texttt{null} if the status of this solution instance is \texttt{NoSolutionValues}.

\textbf{Namespace:} Optimization
\textbf{Assembly:} Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
</tr>
</thead>
<tbody>
<tr>
<td>public Nullable&lt;double&gt; GetVariableValue(string name)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual Basic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Function GetVariableValue (_name As String _) As Nullable(Of Double)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>public: Nullable&lt;double&gt; GetVariableValue( String^ name )</td>
</tr>
</tbody>
</table>

### Parameters

- **name**  
  Type: System.String  
  Name of the variable.

### Return Value

[Missing <returns> documentation for  
"M:Optimization.Solution.GetVariableValue(System.String)""]
See Also

Solution Class
Optimization Namespace
The Solution type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConflictingSet</td>
<td>Gets the conflicting set.</td>
</tr>
<tr>
<td>DualVariableValues</td>
<td>Unique name and solution value for each dual variable in the solved model instance, or <code>null</code> if the status of this solution instance is <code>NoSolutionValues</code>.</td>
</tr>
<tr>
<td>ModelName</td>
<td>Gets the name of the model.</td>
</tr>
<tr>
<td>ModelStatus</td>
<td>Gets the model status.</td>
</tr>
<tr>
<td>ObjectiveValues</td>
<td>Unique name and solution value for each objective in the solved model instance, or <code>null</code> if the status of this solution instance is <code>NoSolutionValues</code>.</td>
</tr>
<tr>
<td>OverallWallTime</td>
<td>Gets the overall wall time.</td>
</tr>
<tr>
<td>Status</td>
<td>Status of this solution instance.</td>
</tr>
<tr>
<td>VariableValues</td>
<td>Unique name and solution value for each variable in the solved model instance, or <code>null</code> if the status of this solution instance is <code>NoSolutionValues</code>.</td>
</tr>
</tbody>
</table>
See Also

Solution Class
Optimization Namespace
Optimization Framework

Solution.ConflictingSet Property

Gets the conflicting set.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#**   | ```
public ConflictingSet ConflictingSet { get; }
``` |
| **Visual Basic** | ```
Public ReadOnly Property ConflictingSet As ConflictingSet
Get
``` |
| **Visual C++**  | ```
public:
property ConflictingSet^ ConflictingSet { 
ConflictingSet^ get ();
}
``` |
- See Also

  Solution Class
  Optimization Namespace
Optimization Framework

Solution.DualVariableValues Property

Unique name and solution value for each dual variable in the solved model instance, or `null` if the status of this solution instance is `NoSolutionValues`.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
</tr>
<tr>
<td>public <code>IDictionary&lt;string, double&gt;</code> DualVariableValues;</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
</tr>
</tbody>
</table>
| Public ReadOnly Property DualVariableValues As `IDictionary<
| Get |
| **Visual C++**  |
| public: `IDictionary<String^, double>`^ DualVariableValues; |
| `IDictionary<String^, double>`^ get (); |
See Also

Solution Class
Optimization Namespace
Solution.ModelName Property

Gets the name of the model.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public string ModelName { get; }</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public ReadOnly Property ModelName As String Get</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public: property String^ ModelName { String^ get (); }</code></td>
</tr>
</tbody>
</table>

### Field Value

The name of the model.
See Also

Solution Class
Optimization Namespace
Gets the model status.

**Namespace:** [Optimization](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
## Syntax

### C#

```csharp
public ModelStatus ModelStatus 
{	get; 
}
```

### Visual Basic

```vbnet
Public ReadOnly Property ModelStatus As ModelStatus
Get
```

### Visual C++

```cpp
public:
property ModelStatus ModelStatus 
{
ModelStatus get ();
}
```

## Field Value

The model status.
See Also

Solution Class
Optimization Namespace
Unique name and solution value for each objective in the solved model instance, or `null` if the status of this solution instance is `NoSolutionValues`.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
</table>
| C#       | ```
public IDictionary<string, double> ObjectiveValues {
``` |
| Visual Basic | ```
Public ReadOnly Property ObjectiveValues As IDictionary
Get
``` |
| Visual C++ | ```
public:
property IDictionary<String^, double>^ ObjectiveValues
IDictionary<String^, double>^ get ();
``` |
See Also

Solution Class
Optimization Namespace
Solution.OverallWallTime Property

Gets the overall wall time.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td><code>public TimeSpan OverallWallTime { get; }</code></td>
</tr>
<tr>
<td>Visual Basic</td>
<td><code>Public ReadOnly Property OverallWallTime As TimeSpan Get</code></td>
</tr>
<tr>
<td>Visual C++</td>
<td><code>public: property TimeSpan OverallWallTime { TimeSpan get (); }</code></td>
</tr>
</tbody>
</table>

## Field Value

The overall wall time.
See Also

Solution Class
Optimization Namespace
Status of this solution instance.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public SolutionStatus Status { get; }
```

**Visual Basic**

```vbnet
PublicReadOnly Property Status As SolutionStatus
    Get
```

**Visual C++**

```cpp
public:
    property SolutionStatus Status {
        SolutionStatus get ();
    }
```
See Also

Solution Class
Optimization Namespace
Unique name and solution value for each variable in the solved model instance, or `null` if the status of this solution instance is `NoSolutionValues`.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>public IDictionary&lt;string, double&gt; VariableValues { get; }</td>
</tr>
</tbody>
</table>
| Visual Basic | Public ReadOnly Property VariableValues As IDictionary<
|            | String, double>^ Get                                                 |
| Visual C++ | public: IDictionary<String^, double>^ VariableValues:
|            | IDictionary<String^, double>^ get ();                               |
See Also

Solution Class
Optimization Namespace
SolverConfiguration Class

Defines common parameters for ISolver instances.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
<tr>
<td><code>public abstract class SolverConfiguration</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td></td>
</tr>
<tr>
<td><code>Public MustInherit Class SolverConfiguration</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
</tr>
<tr>
<td><code>public ref class SolverConfiguration abstract</code></td>
<td></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
Optimization.SolverConfiguration
  Optimization.Solver.MOPS.MOPSSolverConfiguration
See Also

SolverConfiguration Members
Optimization Namespace
The `SolverConfiguration` type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SolverConfiguration()</td>
<td>Initializes a new instance of the SolverConfiguration class.</td>
</tr>
<tr>
<td>SolverConfiguration(String, Boolean)</td>
<td>Initializes a new instance of the SolverConfiguration class with the specified values.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals</td>
<td>Determines whether the specified <a href="Object">Object</a> is equal to the current <a href="Object">Object</a>. (Inherited from <a href="Object">Object</a>.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <a href="Object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="Object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="Object">Object</a>.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="Object">Object</a>.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <a href="Type">Type</a> of the current instance. (Inherited from <a href="Object">Object</a>.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <a href="Object">Object</a>. (Inherited from <a href="Object">Object</a>.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <a href="String">String</a> that represents the current <a href="Object">Object</a>. (Inherited from <a href="Object">Object</a>.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CallbackEndpoint</td>
<td>This endpoint will be used to enable communication between solvers</td>
</tr>
<tr>
<td>GetGlobalIncumbent</td>
<td></td>
</tr>
<tr>
<td>NewBestBoundFound</td>
<td>This action is executed whenever a new best bound is found</td>
</tr>
<tr>
<td>NewIncumbentFound</td>
<td>This action is executed whenever a new incumbent is found</td>
</tr>
<tr>
<td>PartialProblemsNeeded</td>
<td>A method that decides if a partial problem should be created.</td>
</tr>
<tr>
<td>SendPartialProblems</td>
<td>A method that sends partial problems over the wire</td>
</tr>
<tr>
<td>UseHeuristicCallback</td>
<td>Indicates if the solver should activate its provided HeuristicCallback This will make the solver use injected solutions from other solvers</td>
</tr>
</tbody>
</table>
See Also

SolverConfiguration Class
Optimization Namespace
Optimization Framework

**SolverConfiguration Constructor**

[SolverConfiguration Class] [See Also] [Send Feedback]
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SolverConfiguration()</code></td>
<td>Initializes a new instance of the <code>SolverConfiguration</code> class.</td>
</tr>
<tr>
<td><code>SolverConfiguration(String, Boolean)</code></td>
<td>Initializes a new instance of the <code>SolverConfiguration</code> class with the specified values.</td>
</tr>
</tbody>
</table>
See Also

SolverConfiguration Class
SolverConfiguration Members
Optimization Namespace
Initializes a new instance of the `SolverConfiguration` class.

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
<tr>
<td><code>protected SolverConfiguration()</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td></td>
</tr>
<tr>
<td><code>Protected Sub New</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
</tr>
<tr>
<td><code>protected: SolverConfiguration()</code></td>
<td></td>
</tr>
</tbody>
</table>
See Also

SolverConfiguration Class
SolverConfiguration Overload
Optimization Namespace
Optimization Framework

SolverConfiguration Constructor (String, Boolean)

Initializes a new instance of the SolverConfiguration class with the specified values.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
protected SolverConfiguration(
    string callbackendpoint,
    bool useHeuristicCallback
)
```

Visual Basic

```vbnet
Protected Sub New ( _
    callbackendpoint As String, _
    useHeuristicCallback As Boolean _
)
```

Visual C++

```cpp
protected:
SolverConfiguration(
    String^ callbackendpoint,
    bool useHeuristicCallback
)
```

Parameters

`callbackendpoint`
Type: `System.String`
The callback endpoint.

`useHeuristicCallback`
Type: `System.Boolean`
if set to `true` [use heuristic callback].
See Also

SolverConfiguration Class
SolverConfiguration Overload
Optimization Namespace
The `SolverConfiguration` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is equal to the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.type">Type</a> of the current instance. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="https://docs.microsoft.com/en-us/dotnet/api/system.string">String</a> that represents the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

SolverConfiguration Class
Optimization Namespace
The `SolverConfiguration` type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CallbackEndpoint</td>
<td>This endpoint will be used to enable communication between solvers</td>
</tr>
<tr>
<td>GetGlobalIncumbent</td>
<td></td>
</tr>
<tr>
<td>NewBestBoundFound</td>
<td>This action is executed whenever a new best bound is found</td>
</tr>
<tr>
<td>NewIncumbentFound</td>
<td>This action is executed whenever a new incumbent is found</td>
</tr>
<tr>
<td>PartialProblemsNeeded</td>
<td>A method that decides if a partial problem should be created.</td>
</tr>
<tr>
<td>SendPartialProblems</td>
<td>A method that sends partial problems over the wire</td>
</tr>
<tr>
<td>UseHeuristicCallback</td>
<td>Indicates if the solver should activate its provided HeuristicCallback. This will make the solver use injected solutions from other solvers</td>
</tr>
</tbody>
</table>
See Also

SolverConfiguration Class
Optimization Namespace
This endpoint will be used to enable communication between solvers

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>public string CallbackEndpoint { get; private set; }</td>
</tr>
</tbody>
</table>
| Visual Basic | Public Property CallbackEndpoint As String  
  Get  
  Private Set |
| Visual C++ | public:  
  property String^ CallbackEndpoint {  
    String^ get ();  
    private: void set (String^ value);  
  } |
See Also

SolverConfiguration Class
Optimization Namespace

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

**C#**

```csharp
public Func<double> GetGlobalIncumbent { get; set; }
```

**Visual Basic**

```vbnet
Public Property GetGlobalIncumbent As Func(Of Double)
    Get
        '...
    End Get
    Set
        '...
    End Set
```

**Visual C++**

```c++
public:
    property Func<double>^ GetGlobalIncumbent {
        Func<double>^ get ();
        void set (Func<double>^ value);
    }
```
See Also

SolverConfiguration Class
Optimization Namespace
This action is executed whenever a new best bound is found

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
</tr>
</thead>
<tbody>
<tr>
<td>```csharp</td>
</tr>
</tbody>
</table>
| public Action<double> NewBestBoundFound { get; set; }
| ```                   |

<table>
<thead>
<tr>
<th>Visual Basic</th>
</tr>
</thead>
<tbody>
<tr>
<td>```vbnet</td>
</tr>
</tbody>
</table>
| Public Property NewBestBoundFound As Action(Of Double)
| Get |
| Set |
| ```                   |

<table>
<thead>
<tr>
<th>Visual C++</th>
</tr>
</thead>
</table>
| `public:
| property Action<double>^ NewBestBoundFound { 
| Action<double>^ get ();
| void set (Action<double>^ value); 
| }`                   |

### Field Value

The new best bound found.
See Also

SolverConfiguration Class
Optimization Namespace
This action is executed whenever a new incumbent is found

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public <code>Action&lt;double&gt;</code> NewIncumbentFound { get; set; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Property NewIncumbentFound As <code>Action(Of Double)</code> Get Set</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: property <code>Action&lt;double&gt;</code>^ NewIncumbentFound { <code>Action&lt;double&gt;</code>^ get (); void set (<code>Action&lt;double&gt;</code>^ value); }</td>
</tr>
</tbody>
</table>

### Field Value
The new incumbent found.
See Also

SolverConfiguration Class
Optimization Namespace
SolverConfiguration.PartialProblemsNeeded Property

A method that decides if a partial problem should be created.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>public Func&lt;bool&gt; PartialProblemsNeeded { get; set; }</td>
</tr>
<tr>
<td>Visual Basic</td>
<td>Public Property PartialProblemsNeeded As Func(Of Boolean) Get Set</td>
</tr>
<tr>
<td>Visual C++</td>
<td>public: property Func&lt;bool&gt;^ PartialProblemsNeeded { Func&lt;bool&gt;^ get (); void set (Func&lt;bool&gt;^ value); }</td>
</tr>
</tbody>
</table>

### Field Value

The create partial problems.
See Also

SolverConfiguration Class
Optimization Namespace
A method that sends partial problems over the wire

**Namespace:** [Optimization](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

**C#**

```csharp
public Action<IEnumerable<IEnumerable<Constraint>>> SendPartialProblems;
```

**Visual Basic**

```vbnet
Public Property SendPartialProblems As Action(Of IEnumerable)
Get
Set
```

**Visual C++**

```cpp
public:
property Action<IEnumerable<IEnumerable<Constraint>>>^ SendPartialProblems;
void set (Action<IEnumerable<IEnumerable<Constraint>>>^);
```

### Field Value

The send partial problems.
See Also

SolverConfiguration Class
Optimization Namespace
SolverConfiguration.UseHeuristicCallback Property

Indicates if the solver should activate its provided HeuristicCallback. This will make the solver use injected solutions from other solvers.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
</table>
| `public bool UseHeuristicCallback { get; private set;}` | `Public Property UseHeuristicCallback As Boolean
Get
Private Set` | `public:
property bool UseHeuristicCallback {
    bool get ()
    private: void set (bool value);
}` |
See Also

SolverConfiguration Class
Optimization Namespace
The SOS type

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public enum SOSType</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Enumeration SOSType</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public enum class SOSType</code></td>
</tr>
</tbody>
</table>
## Members

<table>
<thead>
<tr>
<th>Member name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOS1</td>
<td>0</td>
<td>A SOS1 is a set of variables in which at most one variable may be positive at an integer feasible solution.</td>
</tr>
<tr>
<td>SOS2</td>
<td>1</td>
<td>A SOS2 is a set of variables in which at most two variables may be positive at an integer feasible solution, and moreover, any positive variables must be adjacent in the ordering specified by the ordering values.</td>
</tr>
<tr>
<td>SOS3</td>
<td>2</td>
<td>The set of variables appearing in an equation with only binary variables and +1 or -1 coefficients, and a right-hand-side value of 1 - (number of -1 coefficients). This is a special case of SOS1.</td>
</tr>
</tbody>
</table>
See Also

Optimization Namespace
This is a helper class to support efficiently building Expressions from a collection of Terms

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
<tr>
<td><code>public class SumExpressionBuilder</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td></td>
</tr>
<tr>
<td><code>Public Class SumExpressionBuilder</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
</tr>
<tr>
<td><code>public ref class SumExpressionBuilder</code></td>
<td></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
Optimization.SumExpressionBuilder
See Also

SumExpressionBuilder Members
Optimization Namespace
The SumExpressionBuilder type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SumExpressionBuilder</td>
<td>Initializes a new instance of the <em>SumExpressionBuilder</em> class.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add</strong></td>
<td>Adds the specified term.</td>
</tr>
<tr>
<td><strong>Clear</strong></td>
<td>Clears this instance.</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>ToExpression</strong></td>
<td>Converts a list of terms to an expression by summing them up</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
</tbody>
</table>
See Also

SumExpressionBuilder Class
Optimization Namespace
Optimization Framework

SumExpressionBuilder Constructor

See Also Send Feedback

Initializes a new instance of the `SumExpressionBuilder` class.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
</tr>
<tr>
<td>public SumExpressionBuilder()</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
</tr>
<tr>
<td>Public Sub New</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
</tr>
<tr>
<td>public:</td>
</tr>
<tr>
<td>SumExpressionBuilder()</td>
</tr>
</tbody>
</table>
See Also

SumExpressionBuilder Class
Optimization Namespace
The **SumExpressionBuilder** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add</strong></td>
<td>Adds the specified term.</td>
</tr>
<tr>
<td><strong>Clear</strong></td>
<td>Clears this instance.</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an Object to attempt to free resources and perform other cleanup</td>
</tr>
<tr>
<td></td>
<td>operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td><strong>ToExpression</strong></td>
<td>Converts a list of terms to an expression by summing them up</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a String that represents the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
</tbody>
</table>
See Also

SumExpressionBuilder Class
Optimization Namespace
Optimization Framework

SumExpressionBuilder.Add Method

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0

Adds the specified term.
## Syntax

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td></td>
</tr>
</tbody>
</table>
| ```csharp``` | public void Add(
| | Term term |
| `)` |   |
| Visual Basic |   |
| ```vbnet``` | Public Sub Add (_
| | term As Term _ |
| `)` |   |
| Visual C++ |   |
| ```cpp``` | public:
| | void Add(
| | Term^ term |
| `)` |   |

### Parameters

- **term**
  - Type: [Optimization.Term](#)
  - The term.
Contracts

Requires

(Object)term != null

Learn more about contracts
See Also

SumExpressionBuilder Class
Optimization Namespace
Clears this instance.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public void Clear()</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Sub Clear</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: void Clear()</td>
</tr>
</tbody>
</table>
See Also

SumExpressionBuilder Class
Optimization Namespace
Optimization Framework

**SumExpressionBuilder.ToExpression Method**

*SumExpressionBuilder Class  See Also  Send Feedback*

Converts a list of terms to an expression by summing them up

**Namespace:** [Optimization](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
## Syntax

**C#**

```csharp
public Expression ToExpression()
```

**Visual Basic**

```vbasic
Public Function ToExpression As Expression
```

**Visual C++**

```cpp
public:
Expression^ ToExpression()
```

---

### Return Value

the resulting Expression
See Also

SumExpressionBuilder Class
Optimization Namespace
A term represents a variable and a coefficient which belongs to this variable (e.g. 2*x)

**Namespace:** [Optimization](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>[SerializableAttribute]</code> public sealed class Term : Expression</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>&lt;SerializableAttribute&gt;</code> _ Public NotInheritable Class Term _ Inherits Expression</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>[SerializableAttribute]</code> public ref class Term sealed : public Expression</td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
Optimization.Expression
Optimization.Term
See Also

Term Members
Optimization Namespace
The **Term** type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Term</td>
<td>Initializes a new instance of the <strong>Term</strong> class.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Evaluate</strong></td>
<td>Evaluates the expression using the specified variable values. (Inherited from <a href="#">Expression</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents this instance. (Overrides <a href="#">Object.ToString()</a>.)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addition(Double, Term)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Term, Term)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Term, Variable)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Double, Double)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Variable, Term)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Implicit(Variable to Term)</td>
<td>Performs an implicit conversion from Variable to Term.</td>
</tr>
<tr>
<td>Multiply(Double, Term)</td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td>Multiply(Term, Term)</td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td>Multiply(Term, Variable)</td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td>Multiply(Double, Double)</td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td>Multiply(Variable, Term)</td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td>Subtraction(Double, Term)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Term, Term)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Term, Variable)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Term, Double)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Variable, Term)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>UnaryNegation</td>
<td>Implements the operator -.</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant</td>
<td>Gets the constant of this expression. (Inherited from Expression.)</td>
</tr>
<tr>
<td>ExpressionLowerEstimate</td>
<td>Gives a lower estimate on the value of an Expression. ToDo: Very experimental. Should be able to estimate linear combinations of variables with finite bounds. In the default case -infinity is returned! (Inherited from Expression.)</td>
</tr>
<tr>
<td>ExpressionUpperEstimate</td>
<td>Gives an upper estimate in the value of an Expression. ToDo: Very experimental. Should be able to estimate linear combinations of variables with finite bounds. In the default case -infinity is returned! (Inherited from Expression.)</td>
</tr>
<tr>
<td>Factor</td>
<td>Gets the factor.</td>
</tr>
<tr>
<td>isLinear</td>
<td>Gets or sets a value indicating whether this instance is linear. (Overrides Expression.isLinear.)</td>
</tr>
<tr>
<td>Terms</td>
<td>Gets the terms. (Overrides Expression.Terms.)</td>
</tr>
<tr>
<td>Variable</td>
<td>Gets the variable.</td>
</tr>
<tr>
<td>Variables</td>
<td>Gets the variables. (Overrides Expression.Variables.)</td>
</tr>
</tbody>
</table>
See Also

Term Class
Optimization Namespace
Initializes a new instance of the **Term** class.

**Namespace:** [Optimization](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

**C#**

```csharp
public Term(IVariable variable, double factor)
```

**Visual Basic**

```vbnet
Public Sub New(_
    variable As IVariable, _
    factor As Double _
)
```

**Visual C++**

```cpp
public:
Term(_
    IVariable^ variable, _
    double factor
)
```

### Parameters

**variable**
Type: `Optimization.Interfaces.IVariable`
The variable.

**factor**
Type: `System.Double`
The constant.
## Exceptions

<table>
<thead>
<tr>
<th>Exception</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>System.ArgumentNullException</td>
<td></td>
</tr>
</tbody>
</table>
See Also

Term Class
Optimization Namespace
The **Term** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>Evaluate</strong></td>
<td>Evaluates the expression using the specified variable values. (Inherited from <strong>Expression</strong>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <strong>Type</strong> of the current instance. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <strong>String</strong> that represents this instance. (Overrides <strong>Object.ToString()</strong>.)</td>
</tr>
</tbody>
</table>
See Also

Term Class
Optimization Namespace
Returns a `String` that represents this instance.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>public override string ToString()</code></td>
<td><code>Public Overrides Function ToString As String</code></td>
<td><code>public: virtual String^ ToString() override</code></td>
</tr>
</tbody>
</table>

**Return Value**

A `String` that represents this instance.
Contracts

Ensures

Contract.Result<string>() != null

Inherited Object

From:

Learn more about contracts
See Also

Term Class
Optimization Namespace
The *Term* type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addition(Double, Term)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Term, Term)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Term, Variable)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Term, Double)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Variable, Term)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Implicit(Variable to Term)</td>
<td>Performs an implicit conversion from Variable to Term.</td>
</tr>
<tr>
<td>Multiply(Double, Term)</td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td>Multiply(Term, Term)</td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td>Multiply(Term, Variable)</td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td>Multiply(Term, Double)</td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td>Multiply(Variable, Term)</td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td>Subtraction(Double, Term)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Term, Term)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Term, Variable)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Term, Double)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Variable, Term)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>UnaryNegation</td>
<td>Implements the operator -.</td>
</tr>
</tbody>
</table>
See Also

Term Class
Optimization Namespace
Optimization Framework

Term.Addition Operator

See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addition(Double, Term)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Term, Term)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Term, Variable)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Term, Double)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Variable, Term)</td>
<td>Implements the operator +.</td>
</tr>
</tbody>
</table>
See Also

Term Class
Term Members
Optimization Namespace
Optimization Framework

Term.Addition Operator (Double, Term)

Implements the operator +.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Expression operator +(  
    double constant,  
    Term term
)
```

**Visual Basic**

```vbscript
Public Shared Operator +(  
    constant As Double,  
    term As Term _
) As Expression
```

**Visual C++**

```cpp
public:  
static Expression^ operator +(  
    double constant,  
    Term^ term
)
```

### Parameters

- **constant**
  - Type: `System.Double`
  - The constant.

- **term**
  - Type: `Optimization.Term`
  - The term.

### ReturnValue
The result of the operator.
<table>
<thead>
<tr>
<th><strong>Contracts</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Requires</strong></td>
</tr>
<tr>
<td>(Object)term != null</td>
</tr>
<tr>
<td><strong>Ensures</strong></td>
</tr>
<tr>
<td>(Object)Contract.Result&lt;Expression&gt;() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Term Class
Addition Overload
Optimization Namespace
Implement the operator `+`.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```
public static Expression operator +(  
    Term term, 
    Term term2
)
```

**Visual Basic**

```
Public Shared Operator + ( _
    term As Term, _
    term2 As Term _
) As Expression
```

**Visual C++**

```
public: 
static Expression^ operator +(  
    Term^ term, 
    Term^ term2
)
```

### Parameters

**term**  
Type: Optimization.Term  
The first term.

**term2**  
Type: Optimization.Term  
The second term.

### Return Value
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)term != null</td>
</tr>
<tr>
<td>(Object)term2 != null</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;Expression&gt;() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Term Class
Addition Overload
Optimization Namespace
Term.Addition Operator (Term, Variable)

Implements the operator +.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Expression operator +(  
    Term term,
    Variable variable
)
```

**Visual Basic**

```vbnet
Public Shared Operator + ( _
    term As Term, _
    variable As Variable _
) As Expression
```

**Visual C++**

```cpp
public:
static Expression^ operator +(  
    Term^ term,
    Variable^ variable
)
```

### Parameters

**term**
Type: **Optimization.Term**
The term.

**variable**
Type: **Optimization.Variable**
The variable.

### Return Value
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!variable.Equals(null)</td>
</tr>
<tr>
<td>(Object)term != null</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;Expression&gt;() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Term Class
Addition Overload
Optimization Namespace
Term.Addition Operator (Term, Double)

Implements the operator `+`.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public static Expression operator +(Term term,
    double constant
)
```

### Visual Basic

```vbnet
Public Shared Operator + (_
    term As Term, _
    constant As Double _
) As Expression
```

### Visual C++

```cpp
public:
    static Expression^ operator +(Term^ term,
        double constant
    )
```

## Parameters

- **term**
  - Type: `Optimization.Term`
  - The term.

- **constant**
  - Type: `System.Double`
  - The constant.

## Return Value
The result of the operator.
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)term != null</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;Expression&gt;() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Term Class
Addition Overload
Optimization Namespace
Optimization Framework

Term.Addition Operator (Variable, Term)

Implements the operator +.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Parameters

**variable**
Type: `Optimization.Variable`
The variable.

**term**
Type: `Optimization.Term`
The term.

### Return Value
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>!variable.Equals(null)</code></td>
</tr>
<tr>
<td><code>(Object)term != null</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>(Object)Contract.Result&lt;Expression&gt;() != null</code></td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Term Class
Addition Overload
Optimization Namespace
Term Implicit Conversion (Variable to Term)

Performs an implicit conversion from Variable to Term.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public static implicit operator Term (Variable variable)
```

### Visual Basic

```vbnet
Public Shared Widening Operator CType (_, variable As Variable _) As Term
```

### Visual C++

```cpp
static implicit operator Term^ (Variable^ variable)
```

## Parameters

*variable*

Type: `Optimization.Variable`

The variable.

## Return Value

The result of the conversion.
See Also

Term Class
Optimization Namespace
Optimization Framework

Term. Multiply Operator

Term Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiply(Double, Term)</td>
<td>Implements the operator *</td>
</tr>
<tr>
<td>Multiply(Term, Term)</td>
<td>Implements the operator *</td>
</tr>
<tr>
<td>Multiply(Term, Variable)</td>
<td>Implements the operator *</td>
</tr>
<tr>
<td>Multiply(Term, Double)</td>
<td>Implements the operator *</td>
</tr>
<tr>
<td>Multiply(Variable, Term)</td>
<td>Implements the operator *</td>
</tr>
</tbody>
</table>
See Also

Term Class
Term Members
Optimization Namespace
Implements the operator `*`.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public static Term operator *(
    double factor,
    Term term
)
```

### Visual Basic

```basic
Public Shared Operator *( _
    factor As Double, _
    term As Term _
) As Term
```

### Visual C++

```cpp
public:
static Term^ operator *(
    double factor,
    Term^ term
)
```

## Parameters

- **factor**
  - Type: `System.Double`
  - The factor.

- **term**
  - Type: `Optimization.Term`
  - The term.

## Return Value
The result of the operator.
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!term.Equals(null)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;Term&gt;() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Term Class
Multiply Overload
Optimization Namespace
Optimization Framework

Term.Multiply Operator (Term, Term)

Implements the operator *.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Expression operator *(  
    Term term,  
    Term term2
)
```

**Visual Basic**

```vbnet
Public Shared Operator *(  
    _  
    term As Term, _  
    term2 As Term _
) As Expression
```

**Visual C++**

```cpp
public:  
static Expression^ operator *(  
    Term^ term,  
    Term^ term2
)
```

### Parameters

- **term**
  - Type: `Optimization.Term`
  - The term.

- **term2**
  - Type: `Optimization.Term`
  - The term2.

### Return Value
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)term != null</td>
<td></td>
</tr>
<tr>
<td>(Object)term2 != null</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
<th></th>
</tr>
</thead>
</table>
| (Object)Contract.Result<
Expression>() != null       |                              |
See Also

Term Class
Multiply Overload
Optimization Namespace
Term.Multiply Operator (Term, Variable)

Implements the operator *.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public static Expression operator *(  
    Term term,  
    Variable variable
)
```

Visual Basic

```vb
Public Shared Operator *(  
    term As Term, _  
    variable As Variable _
) As Expression
```

Visual C++

```cpp
public:  
    static Expression^ operator *(  
        Term^ term,  
        Variable^ variable
    )
```

Parameters

term
Type: Optimization.Term
The term.

variable
Type: Optimization.Variable
The variable.

Return Value
The result of the operator.
**Contracts**

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!variable.Equals(null)</td>
</tr>
<tr>
<td>(Object)term != null</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;Expression&gt;() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Term Class
Multiply Overload
Optimization Namespace
Term.Multiply Operator (Term, Double)

Implements the operator *.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

**C#**

```csharp
public static Term operator *(Term term, double factor)
```

**Visual Basic**

```vbnet
Public Shared Operator *( term As Term, factor As Double ) As Term
```

**Visual C++**

```cpp
public:
static Term^ operator *( Term^ term, double factor)
```

**Parameters**

- `term`
  - Type: `Optimization.Ter`
  - The term.

- `factor`
  - Type: `System.Double`
  - The factor.

**Return Value**
The result of the operator.
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>!object.Equals(term, null)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;Term&gt;() != null</td>
<td></td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Term Class
Multiply Overload
Optimization Namespace
Term.Multiply Operator (Variable, Term)

Implements the operator `*`.

**Namespace:** [Optimization](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
**Syntax**

**C#**

```csharp
public static Expression operator *( 
Variable variable, 
Term term 
)
```

**Visual Basic**

```vbnet
Public Shared Operator *( 
    variable As Variable, 
    term As Term 
) As Expression
```

**Visual C++**

```cpp
public:
static Expression^ operator *( 
Variable^ variable, 
Term^ term 
)
```

**Parameters**

*variable*
Type: Optimization Variable
The variable.

*term*
Type: Optimization Term
The term.

**Return Value**
The result of the operator.
Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!variable.Equals(null)</td>
</tr>
<tr>
<td>(Object)term != null</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;Expression&gt;(() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Term Class
Multiply Overload
Optimization Namespace
Optimization Framework

**Term.Subtraction Operator**

[Term Class See Also Send Feedback]
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subtraction(Double, Term)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Term, Term)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Term, Variable)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Term, Double)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Variable, Term)</td>
<td>Implements the operator -.</td>
</tr>
</tbody>
</table>
See Also

Term Class
Term Members
Optimization Namespace
Implements the operator `-.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
# Syntax

**C#**

```csharp
public static Expression operator -(double constant, Term term)
```

**Visual Basic**

```vbscript
Public Shared Operator - ( constant As Double, term As Term ) As Expression
```

**Visual C++**

```cpp
public:
static Expression^ operator -(double constant, Term^ term)
```

## Parameters

- **constant**
  - Type: `System.Double`
  - The constant.

- **term**
  - Type: `Optimization.Term`
  - The term.

## Return Value
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>(Object)term != null</code></td>
<td><code>(Object)Contract.Result&lt;Expression&gt;() != null</code></td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Term Class
Subtraction Overload
Optimization Namespace
Implements the operator `-.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public static Expression operator -( Term term,
                                  Term term2
)
```

#### Visual Basic

```visualbasic
Public Shared Operator - ( _
    term As Term, _
    term2 As Term _
) As Expression
```

#### Visual C++

```cpp
public:
static Expression^ operator -( Term^ term,
                                Term^ term2
)
```

### Parameters

- **term**
  - Type: **Optimization.Term**
  - The term.

- **term2**
  - Type: **Optimization.Term**
  - The term2.

### Return Value
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)term != null</td>
</tr>
<tr>
<td>(Object)term2 != null</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;Expression&gt;(() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Term Class
Subtraction Overload
Optimization Namespace
Optimization Framework

Term.Subtraction Operator (Term, Variable)

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0

Implements the operator -. 
## Syntax

### C#

```csharp
public static Expression operator -(Term term,
    Variable variable)
```

### Visual Basic

```vbnet
Public Shared Operator -( _
    term As Term, _
    variable As Variable _
) As Expression
```

### Visual C++

```cpp
public:
    static Expression^ operator -(Term^ term,
        Variable^ variable
    )
```

## Parameters

- **term**
  Type: Optimization.Term
  The term.

- **variable**
  Type: Optimization.Variable
  The variable.

## Return Value
The result of the operator.
## Contracts

**Requires**

<table>
<thead>
<tr>
<th>!variable.Equals(null)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)term != null</td>
</tr>
</tbody>
</table>

**Ensures**

| (Object)Contract.Result<Expression>() != null |

Learn more about contracts
See Also

Term Class
Subtraction Overload
Optimization Namespace
Implements the operator -.  

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public static Expression operator -(Term term,
    double constant)
```

Visual Basic

```vbnet
Public Shared Operator -( _
    term As Term, _
    constant As Double _
) As Expression
```

Visual C++

```cpp
public:
    static Expression^ operator -( Term^ term,
        double constant)
```

Parameters

*term*
Type: `Optimization.Term`
The term.

*constant*
Type: `System.Double`
The constant.

Return Value
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)term != null</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;Expression&gt;() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Term Class
Subtraction Overload
Optimization Namespace
Term.Subtraction Operator (Variable, Term)

Implements the operator -.  

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```
public static Expression operator -(  
    Variable variable,  
    Term term  
)
```

#### Visual Basic

```
Public Shared Operator -(  
    variable As Variable,  
    term As Term  
) As Expression
```

#### Visual C++

```
public:  
static Expression^ operator -(  
    Variable^ variable,  
    Term^ term  
)
```

### Parameters

- **variable**  
  Type: [Optimization, Variable]  
  The variable.

- **term**  
  Type: [Optimization, Term]  
  The term.

### Return Value
The result of the operator.
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!variable.Equals(null)</td>
</tr>
<tr>
<td>(Object)term != null</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;Expression&gt;() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Term Class
Subtraction Overload
Optimization Namespace
Optimization Framework

Term.UnaryNegation Operator

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0

Implements the operator -. 
### Syntax

**C#**

```csharp
public static Term operator -(Term term)
```

**Visual Basic**

```vbnet
Public Shared Operator - ( _
    term As Term _
) As Term
```

**Visual C++**

```c++
public:
static Term^ operator -(Term^ term)
```

### Parameters

*term*
Type: *Optimization.Term*
The term.

### Return Value
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>!object.Equals(term, null)</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>(Object)Contract.Result&lt;Term&gt;()</code> != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Term Class
Optimization Namespace
The `Term` type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Constant</strong></td>
<td>Gets the constant of this expression. (Inherited from <code>Expression</code>.)</td>
</tr>
<tr>
<td><strong>ExpressionLowerEstimate</strong></td>
<td>Gives a lower estimate on the value of an Expression. ToDo: Very experimental. Should be able to estimate linear combinations of variables with finite bounds. In the default case -infinity is returned! (Inherited from <code>Expression</code>.)</td>
</tr>
<tr>
<td><strong>ExpressionUpperEstimate</strong></td>
<td>Gives an upper estimate in the value of an Expression. ToDo: Very experimental. Should be able to estimate linear combinations of variables with finite bounds. In the default case -infinity is returned! (Inherited from <code>Expression</code>.)</td>
</tr>
<tr>
<td><strong>Factor</strong></td>
<td>Gets the factor.</td>
</tr>
<tr>
<td><strong>isLinear</strong></td>
<td>Gets or sets a value indicating whether this instance is linear. (Overrides <code>Expression,isLinear</code>.)</td>
</tr>
<tr>
<td><strong>Terms</strong></td>
<td>Gets the terms. (Overides <code>Expression.Terms</code>.)</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>Gets the variable.</td>
</tr>
<tr>
<td><strong>Variables</strong></td>
<td>Gets the variables. (Overides <code>Expression.Variables</code>.)</td>
</tr>
</tbody>
</table>
See Also

Term Class
Optimization Namespace
Gets the factor.

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
</tr>
</thead>
<tbody>
<tr>
<td>public double Factor { get; }</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual Basic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public ReadOnly Property Factor As Double Get</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>public: property double Factor {</td>
</tr>
<tr>
<td>double get ();</td>
</tr>
<tr>
<td>}</td>
</tr>
</tbody>
</table>
See Also

Term Class
Optimization Namespace
Optimization Framework

Term.isLinear Property

Gets or sets a value indicating whether this instance is linear.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public override bool isLinear { get; }
```

**Visual Basic**

```vbnet
Public Overrides ReadOnly Property isLinear As Boolean
Get
```

**Visual C++**

```cpp
public:
virtual property bool isLinear {
    bool get () override;
}
```

### Field Value

`true` if this instance is linear; otherwise, `false`. 
See Also

Term Class
Optimization Namespace
Gets the terms.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public override(IEnumerable&lt;Term&gt;) Terms { get; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Overrides ReadOnly Property Terms As IEnumerable</td>
</tr>
<tr>
<td></td>
<td>Get</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: virtual property IEnumerable&lt;Term&gt;^ Terms { get () override; }</td>
</tr>
</tbody>
</table>
See Also

Term Class
Optimization Namespace
Gets the variable.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public IVariable Variable { get; }</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public ReadOnly Property Variable As IVariable Get</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public: property IVariable^ Variable { IVariable^ get (); }</code></td>
</tr>
</tbody>
</table>
See Also

Term Class
Optimization Namespace
<table>
<thead>
<tr>
<th>Term</th>
<th>Variables Property</th>
</tr>
</thead>
</table>

See Also
Send Feedback

Gets the variables.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public override IEnumerable&lt;IVariable&gt; Variables { get; }</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Overrides ReadOnly Property Variables As IEnumerable&lt;IVariable&gt; Get</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public: virtual property IEnumerable&lt;IVariable&gt;^ Variables;</code></td>
</tr>
</tbody>
</table>
See Also

Term Class
Optimization Namespace
Represents a variable in an `IModel`.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>`[SerializableAttribute] public class Variable : IVariable, ICanRegisterAModel</td>
</tr>
</tbody>
</table>
| Visual Basic      | `<SerializableAttribute> _ Public Class Variable _
                              Implements IVariable, ICanRegisterAModel |
| Visual C++        | `[SerializableAttribute] public ref class Variable : IVariable,
                              ICanRegisterAModel |
Inheritance Hierarchy

System.Object
Optimization.Variable
See Also

Variable Members
Optimization Namespace
The `Variable` type exposes the following members.
### Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable</td>
<td>Initializes a new instance of the Variable class.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Equals(Object)</td>
<td>Determines whether the specified <a href="#">Object</a> is equal to this instance. (Overrides <a href="#">Object.Equals(Object)</a>.)</td>
</tr>
<tr>
<td>Equals(Variable)</td>
<td>Compares this instance to the specified other.</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Returns a hash code for this instance. (Overrides <a href="#">Object.GetHashCode()</a>.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <a href="#">String</a> that represents this instance. (Overrides <a href="#">Object.ToString()</a>.)</td>
</tr>
</tbody>
</table>
## Operators

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Addition(Double, Variable)</code></td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td><code>Addition(Variable, Variable)</code></td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td><code>Addition(Variable, Double)</code></td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td><code>Equality(Variable, Term)</code></td>
<td>Implements the operator ==.</td>
</tr>
<tr>
<td><code>Equality(Variable, Variable)</code></td>
<td>Implements the operator ==.</td>
</tr>
<tr>
<td><code>Equality(Variable, Double)</code></td>
<td>Implements the operator ==.</td>
</tr>
<tr>
<td><code>GreaterThanOrEqual(Variable, Term)</code></td>
<td>Implements the operator &gt;=.</td>
</tr>
<tr>
<td><code>GreaterThanOrEqual(Variable, Double)</code></td>
<td>Implements the operator &gt;=.</td>
</tr>
<tr>
<td><code>Inequality(Variable, Term)</code></td>
<td>Please use &lt;= and &gt;= to model !=.</td>
</tr>
<tr>
<td><code>Inequality(Variable, Variable)</code></td>
<td>Implements the operator !=.</td>
</tr>
<tr>
<td><code>Inequality(Variable, Double)</code></td>
<td>Please use &lt;= and &gt;= to model !=.</td>
</tr>
<tr>
<td><code>LessThanOrEqual(Variable, Term)</code></td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td><code>LessThanOrEqual(Variable, Double)</code></td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td><code>Multiply(Double, Variable)</code></td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td><code>Multiply(Variable, Variable)</code></td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td><code>Multiply(Variable, Double)</code></td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td><code>Subtraction(Double, Variable)</code></td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td><code>Subtraction(Variable, Variable)</code></td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td><code>Subtraction(Variable, Double)</code></td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td><code>UnaryNegation</code></td>
<td>Implements the operator -.</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LowerBound</strong></td>
<td>Lower bound of this variable.</td>
</tr>
<tr>
<td><strong>Name</strong></td>
<td>Unique name of this variable.</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Type of this variable.</td>
</tr>
<tr>
<td><strong>UpperBound</strong></td>
<td>Upper bound of this variable.</td>
</tr>
<tr>
<td><strong>Value</strong></td>
<td>Gets or sets the value for this variable. This field can be used to assign solution values from a solver to the variables in the original model.</td>
</tr>
</tbody>
</table>
See Also

Variable Class
Optimization Namespace
Initializes a new instance of the `Variable` class.

**Namespace:** [Optimization](https://example.com)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public Variable(
    string name,
    double lowerbound,
    double upperbound,
    VariableType type
)
```

#### Visual Basic

```vbnet
Public Sub New (_
    name As String, _
    lowerbound As Double, _
    upperbound As Double, _
    type As VariableType _
)
```

#### Visual C++

```cpp
public:
Variable(
    String^ name,
    double lowerbound,
    double upperbound,
    VariableType type
)
```

### Parameters

- **name**
  - Type: `System.String`
  - The name.
lowerbound
Type: System.Double
The lowerbound.

upperbound
Type: System.Double
The upperbound.

type
Type: Optimization.VariableType
The type.
See Also

Variable Class
Optimization Namespace
The **Variable** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> <em>(Object)</em></td>
<td>Determines whether the specified <em>Object</em> is equal to this instance.</td>
</tr>
<tr>
<td></td>
<td><em>(Overrides <em>Object</em>.Equals(<em>Object)</em>.)</em></td>
</tr>
<tr>
<td><strong>Equals</strong> <em>(Variable)</em></td>
<td>Compares this instance to the specified other.</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <em>Object</em> to attempt to free resources and perform other cleanup operations before the <em>Object</em> is reclaimed by garbage collection. <em>(Inherited from <em>Object</em>.)</em></td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Returns a hash code for this instance. <em>(Overrides <em>Object</em>.GetHashCode()</em>)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <em>Type</em> of the current instance. <em>(Inherited from <em>Object</em>.)</em></td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <em>Object</em>. <em>(Inherited from <em>Object</em>.)</em></td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <em>String</em> that represents this instance. <em>(Overrides <em>Object</em>.ToString()</em>)</td>
</tr>
</tbody>
</table>
See Also

Variable Class
Optimization Namespace
Optimization Framework

Variable.Equals Method

Variable Class See Also Send Feedback
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Equals(Object)</code></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to this instance. (Overrides <code>Object.Equals(Object)</code>.)</td>
</tr>
<tr>
<td><code>Equals(Variable)</code></td>
<td>Compares this instance to the specified other.</td>
</tr>
</tbody>
</table>
See Also

Variable Class
Variable Members
Optimization Namespace
Determines whether the specified `Object` is equal to this instance.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public override bool Equals(
    Object obj
)
```

#### Visual Basic

```vbnet
Public Overrides Function Equals ( _
    obj As Object _
) As Boolean
```

#### Visual C++

```cpp
public:
virtual bool Equals(
    Object^ obj
) override
```

### Parameters

**obj**

Type: `System.Object`

The `Object` to compare with this instance.

### Return Value

`true` if the specified `Object` is equal to this instance; otherwise, `false`. 
See Also

Variable Class
Equals Overload
Optimization Namespace
Variable.Equals Method (Variable)

Compares this instance to the specified other.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public bool Equals(
    Variable other
)
```

#### Visual Basic

```vbnet
Public Function Equals ( _
    other As Variable _
) As Boolean
```

#### Visual C++

```cpp
public:
bool Equals(
    Variable^ other
)
```

### Parameters

*other*

Type: Optimization.Variable

The other.

### Return Value

See Also

Variable Class
Equals Overload
Optimization Namespace
Returns a hash code for this instance.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th></th>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public override int GetHashCode()</td>
<td>Public Overrides Function GetHashCode As Integer</td>
<td>public: virtual int GetHashCode() override</td>
</tr>
</tbody>
</table>

**Return Value**

A hash code for this instance, suitable for use in hashing algorithms and data structures like a hash table.
See Also

Variable Class
Optimization Namespace
Returns a `String` that represents this instance.

**Namespace:** [Optimization](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
**Syntax**

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public override string ToString()</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Overrides Function ToString As String</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public: virtual String^ ToString() override</code></td>
</tr>
</tbody>
</table>

**Return Value**

A `String` that represents this instance.
<table>
<thead>
<tr>
<th><strong>Ensures</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Contract.Result&lt;string&gt;() != null</code></td>
</tr>
</tbody>
</table>

*Inherited from:* `Object`  

Learn more about contracts
See Also

Variable Class
Optimization Namespace
The **Variable** type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addition(Double, Variable)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Variable, Variable)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Variable, Double)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Equality(Variable, Term)</td>
<td>Implements the operator ==.</td>
</tr>
<tr>
<td>Equality(Variable, Variable)</td>
<td>Implements the operator ==.</td>
</tr>
<tr>
<td>Equality(Variable, Double)</td>
<td>Implements the operator ==.</td>
</tr>
<tr>
<td>GreaterThanOrEqual(Variable, Term)</td>
<td>Implements the operator &gt;=.</td>
</tr>
<tr>
<td>GreaterThanOrEqual(Variable, Double)</td>
<td>Implements the operator &gt;=.</td>
</tr>
<tr>
<td>Inequality(Variable, Term)</td>
<td>Please use &lt;= and &gt;= to model !=.</td>
</tr>
<tr>
<td>Inequality(Variable, Variable)</td>
<td>Implements the operator !=.</td>
</tr>
<tr>
<td>Inequality(Variable, Double)</td>
<td>Please use &lt;= and &gt;= to model !=.</td>
</tr>
<tr>
<td>LessThanOrEqual(Variable, Term)</td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td>LessThanOrEqual(Variable, Double)</td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td>Multiply(Double, Variable)</td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td>Multiply(Variable, Variable)</td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td>Multiply(Variable, Double)</td>
<td>Implements the operator *.</td>
</tr>
<tr>
<td>Subtraction(Double, Variable)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Variable, Variable)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Variable, Double)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>UnaryNegation</td>
<td>Implements the operator -.</td>
</tr>
</tbody>
</table>
See Also

Variable Class
Optimization Namespace
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addition(Double, Variable)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Variable, Variable)</td>
<td>Implements the operator +.</td>
</tr>
<tr>
<td>Addition(Variable, Double)</td>
<td>Implements the operator +.</td>
</tr>
</tbody>
</table>
See Also

Variable Class
Variable Members
Optimization Namespace
Variable. Addition Operator (Double, Variable)

Implements the operator +.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Expression operator +(double constant, Variable variable)
```

**Visual Basic**

```vbnet
Public Shared Operator + (_
    constant As Double, _
    variable As Variable_
) As Expression
```

**Visual C++**

```cpp
public:
static Expression^ operator +(double constant,  
    Variable^ variable)
```

### Parameters

- **constant**
  Type: `System.Double`
  The constant.

- **variable**
  Type: `Optimization.Variable`
  The variable.

### Return Value
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
<th>!variable.Equals(null)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ensures</td>
<td>(Object)Contract.Result&lt;Expression&gt;() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Variable Class
Addition Overload
Optimization Namespace
Implements the operator +.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public static Expression operator +(
    Variable variable,
    Variable variable2
)
```

Visual Basic

```vbnet
Public Shared Operator + ( _
    variable As Variable, _
    variable2 As Variable _
) As Expression
```

Visual C++

```cpp
public:
static Expression^ operator +(n
    Variable^ variable,
    Variable^ variable2
)
```

Parameters

- `variable`
  Type: [Optimization, Variable]
  The first variable.

- `variable2`
  Type: [Optimization, Variable]
  The second variable.

Return Value
The result of the operator.
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>!variable.Equals(null)</code></td>
<td></td>
</tr>
<tr>
<td><code>variable2 != null</code></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>(Object)Contract.Result&lt;Expression&gt;() != null</code></td>
<td></td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Variable Class
Addition Overload
Optimization Namespace
Implements the operator `+`.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

<table>
<thead>
<tr>
<th>C#</th>
</tr>
</thead>
<tbody>
<tr>
<td>public static Expression operator +(Variable variable, double constant)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual Basic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Shared Operator + ( variable As Variable, constant As Double ) As Expression</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>public: static Expression ^ operator +(Variable ^ variable, double constant)</td>
</tr>
</tbody>
</table>

**Parameters**

*variable*
- Type: `Optimization,Variable`
- The variable.

*constant*
- Type: `System,Double`
- The constant.

**Return Value**
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!variable.Equals(null)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;Expression&gt;() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts]
See Also

Variable Class
Addition Overload
Optimization Namespace
Variable.Equality Operator
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equality(Variable, Term)</td>
<td>Implements the operator <code>==</code>.</td>
</tr>
<tr>
<td>Equality(Variable, Variable)</td>
<td>Implements the operator <code>==</code>.</td>
</tr>
<tr>
<td>Equality(Variable, Double)</td>
<td>Implements the operator <code>==</code>.</td>
</tr>
</tbody>
</table>
See Also

Variable Class
Variable Members
Optimization Namespace
Variable.Equality Operator (Variable, Term)

Implements the operator ==.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Constraint operator ==(
    Variable variable,
    Term term
)
```

**Visual Basic**

```vbscript
Public Shared Operator = ( _
    variable As Variable, _
    term As Term _
) As Constraint
```

**Visual C++**

```cpp
public:
static Constraint^ operator ==(
    Variable^ variable,
    Term^ term
)
```

### Parameters

- **variable**  
  Type: `Optimization.Variable`  
  The variable.

- **term**  
  Type: `Optimization.Term`  
  The term.

### Return Value
The result of the operator.
## Contracts

### Requires

!variable.Equals(null)

(Object)term != null

### Ensures

Contract.Result<Constraint>() != null

[Learn more about contracts](#)
See Also

Variable Class
Equality Overload
Optimization Namespace
Implements the operator `==`.

**Namespace:** [Optimization](https://example.com/namespace)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static bool operator ==(
    Variable var1,
    Variable var2
)
```

**Visual Basic**

```vbnet
Public Shared Operator = (
    var1 As Variable,
    var2 As Variable
) As Boolean
```

**Visual C++**

```cpp
public:
static bool operator ==(
    Variable^ var1,
    Variable^ var2
)
```

### Parameters

- **var1**
  Type: **Optimization,Variable**
  The first variable.

- **var2**
  Type: **Optimization,Variable**
  The second variable.

### Return Value
The result of the operator.
Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>var1 != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Variable Class
Equality Overload
Optimization Namespace
Optimization Framework

**Variable.Equality Operator (Variable, Double)**

*Variable Class See Also Send Feedback*

Implements the operator `==`.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

**C#**

```csharp
public static Constraint operator ==(Variable variable, double constant)
```

**Visual Basic**

```vbnet
Public Shared Operator = ( _
    variable As Variable, _
    constant As Double _
) As Constraint
```

**Visual C++**

```cpp
public:
static Constraint^ operator ==(Variable^ variable, double constant)
```

**Parameters**

*variable*
Type: `Optimization.Variable`
The variable.

*constant*
Type: `System.Double`
The constant.

**Return Value**
The result of the operator.
### Contracts

**Requires**

!variable.Equals(null)

**Ensures**

Contract.Result<Constraint>(() != null)

[Learn more about contracts](#)
See Also

Variable Class
Equality Overload
Optimization Namespace
Optimization Framework

Variable.GreaterThanOrEqual Operator

Variable Class  See Also  Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>GreaterThanEqual(Variable, Term)</code></td>
<td>Implements the operator <code>&gt;=</code>.</td>
</tr>
<tr>
<td><code>GreaterThanEqual(Variable, Double)</code></td>
<td>Implements the operator <code>&gt;=</code>.</td>
</tr>
</tbody>
</table>
See Also

Variable Class
Variable Members
Optimization Namespace
Implements the operator >=.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public static Constraint operator >=(
    Variable variable,
    Term term
)
```

### Visual Basic

```vbnet
Public Shared Operator >= ( _
    variable As Variable, _
    term As Term _
) As Constraint
```

### Visual C++

```cpp
public:
    static Constraint^ operator >=(
        Variable^ variable,
        Term^ term
    )
```

## Parameters

**variable**
Type: `Optimization.Variable`
The variable.

**term**
Type: `Optimization.Term`
The term.

## Return Value
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!variable.Equals(null)</td>
</tr>
<tr>
<td>(Object)term != null</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contract.Result&lt;Constraint&gt;() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Variable Class
GreaterThanOrEqual Overload
Optimization Namespace
Implements the operator >=.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public static Constraint operator >=(
    Variable variable,
    double constant
)
```

### Visual Basic

```vbnet
Public Shared Operator >= (_
    variable As Variable, _
    constant As Double _
) As Constraint
```

### Visual C++

```cpp
public:
static Constraint^ operator >=(
    Variable^ variable,
    double constant
)
```

## Parameters

- **variable**
  - Type: **Optimization.Variable**
  - The variable.

- **constant**
  - Type: **System.Double**
  - The constant.

## Return Value
The result of the operator.
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!variable.Equals(null)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contract.Result&lt;Constraint&gt;() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Variable Class
GreaterThanOrEqual Overload
Optimization Namespace
Variable.Inequality Operator
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inequality(Variable, Term)</td>
<td>Please use &lt;= and &gt;= to model !=</td>
</tr>
<tr>
<td>Inequality(Variable, Variable)</td>
<td>Implements the operator !=.</td>
</tr>
<tr>
<td>Inequality(Variable, Double)</td>
<td>Please use &lt;= and &gt;= to model !=</td>
</tr>
</tbody>
</table>
See Also

Variable Class
Variable Members
Optimization Namespace
Variable. Inequality Operator (Variable, Term)

Please use <= and >= to model !=

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Constraint operator !=(
    Variable variable,
    Term term
)
```

**Visual Basic**

```vbnet
Public Shared Operator <> ( _
    variable As Variable, _
    term As Term _
) As Constraint
```

**Visual C++**

```cpp
public:
static Constraint^ operator !=(
    Variable^ variable,
    Term^ term
)
```

### Parameters

**variable**
- Type: `Optimization.Variable`
- The variable.

**term**
- Type: `Optimization.Term`
- The term.

### Return Value
The result of the operator.
See Also

Variable Class
Inequality Overload
Optimization Namespace
Optimization Framework

Variable.Inequality Operator (Variable, Variable)

Implements the operator !=.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static bool operator !=(
    Variable var1,
    Variable var2
)
```

**Visual Basic**

```vbnet
Public Shared Operator <> ( _
    var1 As Variable, _
    var2 As Variable _
) As Boolean
```

**Visual C++**

```cpp
public:
    static bool operator !=(
        Variable^ var1,
        Variable^ var2
    )
```

### Parameters

**var1**

Type: `Optimization.Variable`

The first variable.

**var2**

Type: `Optimization.Variable`

The second variable.

### Return Value
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>var1 != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Variable Class
Inequality Overload
Optimization Namespace
Please use <= and >= to model !=

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public static Constraint operator !=(
    Variable variable, double constant
)
```

### Visual Basic

```vbnet
Public Shared Operator <> ( _
    variable As Variable, _
    constant As Double _
) As Constraint
```

### Visual C++

```cpp
public:
    static Constraint^ operator !=(
        Variable^ variable, double constant
    )
```

## Parameters

- **variable**
  - Type: `Optimization.Variable`
  - The variable.

- **constant**
  - Type: `System.Double`
  - The constant.

## Return Value
The result of the operator.
See Also

Variable Class
Inequality Overload
Optimization Namespace
Variable.LessThanOrEqual Operator

Variable Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LessThanOrEqual(Variable, Term)</td>
<td>Implements the operator &lt;=.</td>
</tr>
<tr>
<td>LessThanOrEqual(Variable, Double)</td>
<td>Implements the operator &lt;=.</td>
</tr>
</tbody>
</table>
See Also

Variable Class
Variable Members
Optimization Namespace
Variable.LessThanOrEqual Operator (Variable, Term)

Implements the operator \( \leq \).

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public static Constraint operator <=(
    Variable variable,
    Term term
)
```

### Visual Basic

```vbnet
Public Shared Operator <= ( _
    variable As Variable, _
    term As Term _
) As Constraint
```

### Visual C++

```cpp
public:
    static Constraint^ operator <=(
        Variable^ variable,
        Term^ term
    )
```

## Parameters

- **variable**
  Type: `Optimization.Variable`
  The variable.

- **term**
  Type: `Optimization.Term`
  The term.

## Return Value
The result of the operator.
# Contracts

**Requires**

!variable.Equals(null)

(Object)term != null

**Ensures**

Contract.Result<Constraint>() != null

[Learn more about contracts](#)
See Also

Variable Class
LessThanOrEqual Overload
Optimization Namespace
Implements the operator \( \leq \).

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

### C#

```csharp
public static Constraint operator <=(
    Variable variable,
    double constant
)
```

### Visual Basic

```vbnet
Public Shared Operator <= ( _
    variable As Variable, _
    constant As Double _
) As Constraint
```

### Visual C++

```cpp
public:
    static Constraint^ operator <=(
        Variable^ variable,
        double constant
    )
```

#### Parameters

- **variable**
  Type: **Optimization.Variable**
  The variable.

- **constant**
  Type: **System.Double**
  The constant.

#### Return Value
The result of the operator.
Contracts

**Requires**

!variable.Equals(null)

**Ensures**

Contract.Result<Constraint>() != null

Learn more about contracts
See Also

Variable Class
LessThanOrEqual Overload
Optimization Namespace
Variable.Multiply Operator

Variable Class  See Also  Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiply(Double, Variable)</td>
<td>Implements the operator *</td>
</tr>
<tr>
<td>Multiply(Variable, Variable)</td>
<td>Implements the operator *</td>
</tr>
<tr>
<td>Multiply(Variable, Double)</td>
<td>Implements the operator *</td>
</tr>
</tbody>
</table>
See Also

Variable Class
Variable Members
Optimization Namespace
Optimization Framework

Variable.Multiply Operator (Double, Variable)

Implements the operator *.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

<table>
<thead>
<tr>
<th>C#</th>
</tr>
</thead>
<tbody>
<tr>
<td>public static Term operator *(</td>
</tr>
<tr>
<td>double constant,</td>
</tr>
<tr>
<td>Variable variable</td>
</tr>
<tr>
<td>)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual Basic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Shared Operator * ( _</td>
</tr>
<tr>
<td>constant As Double, _</td>
</tr>
<tr>
<td>variable As Variable _</td>
</tr>
<tr>
<td>) As Term</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>public:</td>
</tr>
<tr>
<td>static Term^ operator *(</td>
</tr>
<tr>
<td>double constant,</td>
</tr>
<tr>
<td>Variable^ variable</td>
</tr>
<tr>
<td>)</td>
</tr>
</tbody>
</table>

**Parameters**

*constant*

Type: System.Double

The constant.

*variable*

Type: Optimization.Variable

The variable.

**Return Value**
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
<th>!variable.Equals(null)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ensures</td>
<td>(Object)Contract.Result&lt;Term&gt;()) != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Variable Class
Multiply Overload
Optimization Namespace
Variable.Multiply Operator (Variable, Variable)

Implements the operator *.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>C#</th>
</tr>
</thead>
</table>
| public static Expression operator *(  
  Variable variable1,  
  Variable variable2  
) |

<table>
<thead>
<tr>
<th>Visual Basic</th>
</tr>
</thead>
</table>
| Public Shared Operator * (  
  _  
  variable1 As Variable, _  
  variable2 As Variable _  
) As Expression |

<table>
<thead>
<tr>
<th>Visual C++</th>
</tr>
</thead>
</table>
| public:  
  static Expression^ operator *(  
  Variable^ variable1,  
  Variable^ variable2  
) |

## Parameters

- **variable1**  
  Type: [Optimization,Variable]  
  The first variable.

- **variable2**  
  Type: [Optimization,Variable]  
  The second variable.

## Return Value
The result of the operator.
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>variable1 != null</td>
</tr>
<tr>
<td>variable2 != null</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;Expression&gt;() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Variable Class
Multiply Overload
Optimization Namespace
Implements the operator `*`.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#
```
public static Term operator *(Variable variable, double constant)
```

### Visual Basic
```
Public Shared Operator * ( _
    variable As Variable, _
    constant As Double _
) As Term
```

### Visual C++
```
public:
static Term^ operator *( 
    Variable^ variable, 
    double constant 
)
```

### Parameters

**variable**
Type: `Optimization.Variable`
The variable.

**constant**
Type: `System.Double`
The constant.

### Return Value
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>!variable.Equals(null)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Object)Contract.Result&lt;Term&gt;() != null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

Variable Class
Multiply Overload
Optimization Namespace
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subtraction(Double, Variable)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Variable, Variable)</td>
<td>Implements the operator -.</td>
</tr>
<tr>
<td>Subtraction(Variable, Double)</td>
<td>Implements the operator -.</td>
</tr>
</tbody>
</table>
See Also

Variable Class
Variable Members
Optimization Namespace
Optimization Framework

Variable.Subtraction Operator (Double, Variable)

Implements the operator -.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

public static Expression operator -( double constant, Variable variable )

Visual Basic

Public Shared Operator - ( _
    constant As Double, _
    variable As Variable _
) As Expression

Visual C++

public:
static Expression^ operator -( double constant, Variable^ variable )

Parameters

constant
Type: System.Double
The constant.

variable
Type: Optimization.Variable
The variable.

Return Value
The result of the operator.
See Also

Variable Class
Subtraction Overload
Optimization Namespace
Variable.Subtraction Operator (Variable, Variable)

Implements the operator `-.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
</tbody>
</table>
| public static Expression operator -(  
  Variable variable,  
  Variable variable2  
) |
| **Visual Basic** |     |
| Public Shared Operator - (  _  
  variable As Variable, _  
  variable2 As Variable _  
) As Expression |
| **Visual C++** |     |
| public: Expression^ operator -(  
  Variable^ variable,  
  Variable^ variable2  
) |

### Parameters

- **variable**
  - Type: Optimization,Variable
  - The first variable.

- **variable2**
  - Type: Optimization,Variable
  - The second variable.

### Return Value
The result of the operator.
## Contracts

### Requires

- `!variable.Equals(null)`
- `variable2 != null`

### Ensures

- `(Object)Contract.Result<Expression>() != null`

[Learn more about contracts](#)
See Also

Variable Class
Subtraction Overload
Optimization Namespace
Implements the operator -.

**Namespace:** [Optimization](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public static Expression operator -(Variable variable, double constant )
```

**Visual Basic**

```vbnet
Public Shared Operator -( _
    variable As Variable, _
    constant As Double _
) As Expression
```

**Visual C++**

```cpp
public: 
static Expression^ operator -( 
    Variable^ variable, 
    double constant 
)
```

### Parameters

**variable**
Type: `Optimization.Variable`
The variable.

**constant**
Type: `System.Double`
The constant.

### Return Value
The result of the operator.
## Contracts

<table>
<thead>
<tr>
<th>Requires</th>
<th>!variable.Equals(null)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ensures</td>
<td>(Object)Contract.Result&lt;Expression&gt;() != null</td>
</tr>
</tbody>
</table>

Learn more about contracts
See Also

Variable Class
Subtraction Overload
Optimization Namespace
Implements the operator «-».

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

**C#**

```csharp
public static Term operator -(Variable variable)
```

**Visual Basic**

```vbnet
Public Shared Operator -( _
    variable As Variable _
) As Term
```

**Visual C++**

```cpp
public:
static Term^ operator -( _
    Variable^ variable
)
```

**Parameters**

*variable*

Type: Optimization, Variable

The variable.

**Return Value**

A term with the factor of -1 and the variable given.
See Also

Variable Class
Optimization Namespace
The **Variable** type exposes the following members.
# Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LowerBound</td>
<td>Lower bound of this variable.</td>
</tr>
<tr>
<td>Name</td>
<td>Unique name of this variable.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of this variable.</td>
</tr>
<tr>
<td>UpperBound</td>
<td>Upper bound of this variable.</td>
</tr>
<tr>
<td>Value</td>
<td>Gets or sets the value for this variable. This field can be used to assign solution values from a solver to the variables in the original model.</td>
</tr>
</tbody>
</table>
See Also

Variable Class
Optimization Namespace
Lower bound of this variable.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>public double LowerBound { get; set; }</code></td>
<td><code>Public Property LowerBound As Double Get Set</code></td>
<td><code>public: virtual property double LowerBound {</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>    double get () sealed;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>    void set (double value) sealed;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>}</code></td>
</tr>
</tbody>
</table>

**Field Value**

**Implements**

[IVariable.LowerBound]
See Also

Variable Class
Optimization Namespace
Unique name of this variable.

**Namespace:** [Optimization](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public <code>string</code> Name { get; set; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Property Name As <code>String</code> Get Set</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: virtual property <code>String</code>^ Name { <code>String</code>^ get () sealed; void set (<code>String</code>^ value) sealed; }</td>
</tr>
</tbody>
</table>

### Field Value

**Implements**

`IVariable.Name`
See Also

Variable Class
Optimization Namespace
Type of this variable.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public VariableType Type { get; set; }
```

**Visual Basic**

```vbnet
Public Property Type As VariableType
    Get
        Type
    Set
        Type
```

**Visual C++**

```cpp
public:
    virtual property VariableType Type {
        VariableType get () sealed;
        void set (VariableType value) sealed;
    }
```

### Field Value

**Implements**

`Variable.Type`
See Also

Variable Class
Optimization Namespace
Upper bound of this variable.

**Namespace:** [Optimization](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
## Syntax

### C#

```csharp
public double UpperBound { get; set; }
```

### Visual Basic

```vbnet
Public Property UpperBound As Double
    Get
        Set
```

### Visual C++

```cpp
public:
    virtual property double UpperBound {
        double get () sealed;
        void set (double value) sealed;
    }
```

## Field Value

Implements

`IVariable.UpperBound`
See Also

Variable Class
Optimization Namespace
Gets or sets the value for this variable. This field can be used to assign solution values from a solver to the variables in the original model.

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

**C#**

```csharp
public double Value { get; set; }
```

**Visual Basic**

```vbnet
Public Property Value As Double
    Get
    Set
```

**Visual C++**

```cpp
public:
    virtual property double Value {
        double get () sealed;
        void set (double value) sealed;
    }
```

### Field Value

The value.

### Implements

`IVariable.Value`
See Also

Variable Class
Optimization Namespace
This is essentially a helper class that allows you to model your Variables more intuitively

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
<tr>
<td><code>public class VariableCollection</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td></td>
</tr>
<tr>
<td><code>Public Class VariableCollection</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
</tr>
<tr>
<td><code>public ref class VariableCollection</code></td>
<td></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object

Optimization.VariableCollection
See Also

VariableCollection Members
Optimization Namespace
The VariableCollection type exposes the following members.
# Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VariableCollection(String, Double, Double, VariableType, IEnumerable[])</td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
<tr>
<td>VariableCollection(VariableCollection.UniqueNameGenerator, VariableCollection.BoundGenerator, VariableCollection.BoundGenerator, VariableType, IEnumerable[])</td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
<tr>
<td>VariableCollection(VariableCollection.UniqueNameGenerator, Double, Double, VariableType, IEnumerable[])</td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CreateAllVariables</strong></td>
<td>Creates all variables that can possibly be created based on the elements currently in the sets.</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index Validation</td>
<td>Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on</td>
</tr>
<tr>
<td>Item</td>
<td>Gets the <a href="#">Variable</a> with the specified index.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection Class
Optimization Namespace
Optimization Framework

VariableCollection Constructor

VariableCollection Class See Also Send Feedback
# Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>VariableCollection(String, Double, Double, VariableType, IEnumerable[])</code></td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
<tr>
<td><code>VariableCollection(VariableCollection, UniqueNameGenerator, VariableCollection, BoundGenerator, VariableCollection, BoundGenerator, VariableType, IEnumerable[])</code></td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
<tr>
<td><code>VariableCollection(VariableCollection, UniqueNameGenerator, Double, Double, VariableType, IEnumerable[])</code></td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection Class
VariableCollection Members
Optimization Namespace
Initializes a new instance of the **VariableCollection** class.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) **Version:** 0.3.1036.0
## Syntax

### C#

```csharp
public VariableCollection(
    string name,
    double lowerBound,
    double upperBound,
    VariableType type,
    params IEnumerable[] sets
)
```

### Visual Basic

```vbnet
Public Sub New (_
    name As String, _
    lowerBound As Double, _
    upperBound As Double, _
    type As VariableType, _
    ParamArray sets As IEnumerable() _
)
```

### Visual C++

```cpp
public:
VariableCollection(
    String^ name,
    double lowerBound,
    double upperBound,
    VariableType type,
    ... array<IEnumerable^>^ sets
)
```

## Parameters
name
Type: System.String
The name.

lowerBound
Type: System.Double
The lower bound.

upperBound
Type: System.Double
The upper bound.

type
Type: Optimization.VariableType
The type.

sets
Type: System.Collections.IEnumerable[]
The sets.
See Also

VariableCollection Class
VariableCollection Overload
Optimization Namespace
VariableCollection Constructor
(VariableCollection.UniqueNameGenerator,
VariableCollection.BoundGenerator, VariableCollection.BoundGenerator,
VariableType, IEnumerable[])

Initializes a new instance of the **VariableCollection** class.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#** | ```
public VariableCollection(
    VariableCollection.UniqueNameGenerator uniqueNamegenerator,
    VariableCollection.BoundGenerator lowerBoundGenerator,
    VariableCollection.BoundGenerator upperBoundGenerator,
    VariableType type,
    params IEnumerable[] sets
)
``` |
| **Visual Basic** | ```
Public Sub New (_
    uniqueNamegenerator As VariableCollection.UniqueNameGenerator,
    lowerBoundGenerator As VariableCollection.BoundGenerator,
    upperBoundGenerator As VariableCollection.BoundGenerator,
    type As VariableType, _
    ParamArray sets As IEnumerable() _
)
``` |
| **Visual C++** | ```
public:
VariableCollection(
    VariableCollection.UniqueNameGenerator^ uniqueNamegenerator,
    VariableCollection.BoundGenerator^ lowerBoundGenerator,
    VariableCollection.BoundGenerator^ upperBoundGenerator,
    VariableType type,
    ... array< IEnumerable^ >^ sets
)
``` |

### Parameters
uniqueNamegenerator
Type: Optimization,VariableCollection,UniqueNameGenerator
A unique namegenerator.

lowerBoundGenerator
Type: Optimization,VariableCollection,BoundGenerator
A generator function for the lower bound

upperBoundGenerator
Type: Optimization,VariableCollection,BoundGenerator
A generator function for the upper bound

type
Type: Optimization,VariableType
The type.

sets
Type: System.Collections,IEnumerable[]
The sets.
See Also

VariableCollection Class
VariableCollection Overload
Optimization Namespace
Initializes a new instance of the **VariableCollection** class.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public VariableCollection(
    VariableCollection.UniqueNameGenerator uniqueNamegenerator,
    double lowerBound,
    double upperBound,
    VariableType type,
    params IEnumerable[] sets
)
```

#### Visual Basic

```vbnet
Public Sub New ( _
    uniqueNamegenerator As VariableCollection.UniqueNameGenerator, _
    lowerBound As Double, _
    upperBound As Double, _
    type As VariableType, _
    ParamArray sets As IEnumerable() _
)
```

#### Visual C++

```cpp
public:
    VariableCollection(
        VariableCollection.UniqueNameGenerator^ uniqueNamegenerator,
        double lowerBound,
        double upperBound,
        VariableType type,
        ... array<IEnumerable^>^ sets
    )
```

### Parameters
**uniqueNamegenerator**
Type: **Optimization,VariableCollection,UniqueNameGenerator**
The unique namegenerator.

**lowerBound**
Type: **System.Double**
The lower bound.

**upperBound**
Type: **System.Double**
The upper bound.

**type**
Type: **Optimization,VariableType**
The type.

**sets**
Type: **System.Collections,IEnumerable[]**
The sets.
See Also

VariableCollection Class
VariableCollection Overload
Optimization Namespace
The **VariableCollection** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CreateAllVariables</strong></td>
<td>Creates all variables that can possibly be created based on the elements currently in the sets.</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>)</td>
</tr>
</tbody>
</table>
See Also

VariableCollection Class
Optimization Namespace
Creates all variables that can possibly be created based on the elements currently in the sets.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public IEnumerable<IVariable> CreateAllVariables()
```

Visual Basic

```vbnet
Public Function CreateAllVariables As IEnumerable(Of VisualC++)
```

Visual C++

```cpp
public:
IEnumerable<IVariable>^ CreateAllVariables()
```

**Return Value**

All variables in this VariableCollection
See Also

VariableCollection Class
Optimization Namespace
Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public void SetVariableValues(
    IDictionary<string, double> values
)
```

### Visual Basic

```vbnet
Public Sub SetVariableValues (_
    values As IDictionary(Of String, Double) _
)
```

### Visual C++

```cpp
public:
    void SetVariableValues(
        IDictionary<String^, double>^ values
    )
```

## Parameters

**values**

Type: `System.Collections.Generic.IDictionary(String, Double)`
The values for the variables (e.g. coming from a solution)
See Also

VariableCollection Class
Optimization Namespace
The VariableCollection type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index Validation</td>
<td>Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on</td>
</tr>
<tr>
<td>Item</td>
<td>Gets the <a href="#">Variable</a> with the specified index.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection Class
Optimization Namespace
VariableCollection.IndexValidation Property

Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
</tr>
<tr>
<td>public <code>bool</code> IndexValidation { get; set; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
</tr>
</tbody>
</table>
| Public Property IndexValidation As `Boolean`  
  Get  
  Set |
| **Visual C++**   |
| public: `bool` IndexValidation {  
  `bool` get ();  
  void set (`bool` value);  
} |
See Also

VariableCollection Class
Optimization Namespace
Gets the Variable with the specified index.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public Variable this[
    params Object[] index
] { get; }
```

**Visual Basic**

```vbnet
Public ReadOnly Default Property Item ( _
    ParamArray index As Object() _
) As Variable
    Get
```

**Visual C++**

```cpp
public:
    property Variable^ default[... array<Object^>^ index]
        Variable^ get (... array<Object^>^ index);
```

### Parameters

**index**

Type: `System.Object[]`

### Field Value
See Also

VariableCollection Class
Optimization Namespace
A delegate which takes an array of objects (making up the index) and returns a bound to be used for a variable

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public delegate double BoundGenerator(
    params Object[] index
)
```

### Visual Basic

```vbnet
Public Delegate Function BoundGenerator (_
    ParamArray index As Object() _) As Double
```

### Visual C++

```cpp
public delegate double BoundGenerator(
    ... array<Object^>^ index
)
```

## Parameters

- **index**
  - Type: `System.Object[]`
  - The index.

## Return Value
See Also

Optimization Namespace
A delegate which takes an array of objects (making up the index) and returns a stringbuilder which holds a unique id for this index

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Delegate Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>`public delegate StringBuilder UniqueNameGenerator(</td>
</tr>
<tr>
<td></td>
<td>params Object[] index)</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>`Public Delegate Function UniqueNameGenerator (</td>
</tr>
<tr>
<td></td>
<td>_ ParamArray index As Object() _) As StringBuilder</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>`public delegate StringBuilder^ UniqueNameGenerator(</td>
</tr>
<tr>
<td></td>
<td>... array&lt;Object^&gt;^ index)</td>
</tr>
</tbody>
</table>

#### Parameters

- **index**
  - Type: `System.Object[]`
  - The index.

#### Return Value

- A unique ID for this index
See Also

Optimization Namespace
VariableCollection\(T\) Class

This is a strongly typed helper class that allows you to model your Variables more intuitively

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
<tr>
<td>public class VariableCollection&lt;T&gt; : GenericVariableCollectionBase</td>
<td></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td></td>
</tr>
<tr>
<td>Public Class VariableCollection(Of T) _ Inherits GenericVariableCollectionBase</td>
<td></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
</tr>
<tr>
<td>generic&lt;typename T&gt; public ref class VariableCollection : public GenericVariableCollectionBase</td>
<td></td>
</tr>
</tbody>
</table>
Type Parameters

\[T\]

[Missing <typeparam name="T"/> documentation for "T:Optimization.VariableCollection`1"]
Inheritance Hierarchy

System.Object
Optimization.GenericVariableCollectionBase
Optimization.VariableCollection(T)
See Also

VariableCollection(T) Members
Optimization Namespace
The **VariableCollection(T)** type exposes the following members.
### Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VariableCollection&lt;T&gt;(Func&lt;T, StringBuilder, Double, Double, VariableType, IEnumerable&lt;T&gt;)</strong></td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
<tr>
<td><strong>VariableCollection&lt;T&gt;(String, Double, Double, VariableType, IEnumerable&lt;T&gt;)</strong></td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="System.Object">Object</a> is equal to the current <a href="System.Object">Object</a>. (Inherited from <a href="System.Object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="System.Object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="System.Object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="System.Object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="System.Object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="System.Type">Type</a> of the current instance. (Inherited from <a href="System.Object">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="System.Object">Object</a>. (Inherited from <a href="System.Object">Object</a>.)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable. (Inherited from <a href="System.Object">GenericVariableCollectionBase</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="System.String">String</a> that represents the current <a href="System.Object">Object</a>. (Inherited from <a href="System.Object">Object</a>.)</td>
</tr>
</tbody>
</table>
### Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>_internalVariableCollection</code></td>
<td>(Inherited from <a href="#">GenericVariableCollectionBase</a>.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Icon" /> <strong>Index Validation</strong></td>
<td>Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on (Inherited from <a href="#">GenericVariableCollectionBase</a>.)</td>
</tr>
<tr>
<td><img src="image" alt="Icon" /> <strong>Item</strong></td>
<td>Gets the <a href="#">Variable</a> with the specified index1.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T) Class
Optimization Namespace
Optimization Framework

VariableCollection\((T)\) Constructor

VariableCollection\((T)\) Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VariableCollection(T)(Func(T, StringBuilder, Double, Double, VariableType, IEnumerable(T)))</td>
<td>Initializes a new instance of the VariableCollection class.</td>
</tr>
<tr>
<td>VariableCollection(T)(String, Double, Double, VariableType, IEnumerable(T))</td>
<td>Initializes a new instance of the VariableCollection class.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T) Class
VariableCollection(T) Members
Optimization Namespace
VariableCollection(T) Constructor (Func(T, StringBuilder), Double, Double, VariableType, IEnumerable(T))

Initializes a new instance of the VariableCollection class.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public VariableCollection(
    Func<T, StringBuilder> uniqueNamegenerator,
    double lowerBound,
    double upperBound,
    VariableType type,
    IEnumerable<T> set1
)
```

**Visual Basic**

```vbnet
Public Sub New (_
    uniqueNamegenerator As Func(Of T, StringBuilder), _
    lowerBound As Double, _
    upperBound As Double, _
    type As VariableType, _
    set1 As IEnumerable(Of T) _
)
```

**Visual C++**

```cpp
public:
VariableCollection(
    Func<T, StringBuilder>^ uniqueNamegenerator,
    double lowerBound,
    double upperBound,
    VariableType type,
    IEnumerable<T>^ set1
)
```

### Parameters
**uniqueNamegenerator**
Type: `System.Func<T, StringBuilder>`
A unique namegenerator.

**lowerBound**
Type: `System.Double`
The lower bound.

**upperBound**
Type: `System.Double`
The upper bound.

**type**
Type: `Optimization.VariableType`
The type.

**set1**
Type: `System.Collections.Generic.IEnumerable<T>`
The first set
See Also

VariableCollection(T) Class
VariableCollection(T) Overload
Optimization Namespace
Optimization Framework

VariableCollection(T) Constructor (String, Double, Double, VariableType, IEnumerable(T))

VariableCollection(T) Class See Also Send Feedback

Initializes a new instance of the VariableCollection class.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public VariableCollection(
    string name,
    double lowerBound,
    double upperBound,
    VariableType type,
    IEnumerable<T> set1
)
```

#### Visual Basic

```vbnet
Public Sub New (_
    name As String, _
    lowerBound As Double, _
    upperBound As Double, _
    type As VariableType, _
    set1 As IEnumerable(Of T) _
)
```

#### Visual C++

```cpp
public:
VariableCollection(
    String^ name,
    double lowerBound,
    double upperBound,
    VariableType type,
    IEnumerable<T>^ set1
)
```

### Parameters
name
Type: System.String
The name.

lowerBound
Type: System.Double
The lower bound.

upperBound
Type: System.Double
The upper bound.

type
Type: Optimization.VariableType
The type.

set1
Type: System.Collections.Generic.IEnumerable(T)
The first set
See Also

VariableCollection(T) Class
VariableCollection(T) Overload
Optimization Namespace
Optimization Framework

**VariableCollection(T) Fields**

The `VariableCollection<T>` type exposes the following members.
### Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_internalVariableCollection</td>
<td>(Inherited from <a href="#">GenericVariableCollectionBase</a>)</td>
</tr>
</tbody>
</table>
See Also

VariableCollection<T> Class
Optimization Namespace
The `VariableCollection<T>` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <strong>Type</strong> of the current instance. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable. (Inherited from <strong>GenericVariableCollectionBase</strong>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T) Class
Optimization Namespace
Optimization Framework

VariableCollection(T) Properties

VariableCollection(T) Class  See Also  Send Feedback

The VariableCollection(T) type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index Validation</td>
<td>Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on (Inherited from <code>GenericVariableCollectionBase</code>).</td>
</tr>
<tr>
<td>Item</td>
<td>Gets the Variable with the specified index1.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T) Class
Optimization Namespace
Gets the `Variable` with the specified index1.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public Variable this[
    T element1
] { get; }
```

### Visual Basic

```vbnet
Public Readonly Default Property Item ( _
    element1 As T _
) As Variable
    Get
```

### Visual C++

```cpp
public:
    property Variable^ default[T element1] { 
        Variable^ get (T element1);
    }
```

## Parameters

*element1*

Type: `T`
See Also

VariableCollection(T) Class
Optimization Namespace
This is a strongly typed helper class that allows you to model your Variables more intuitively.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public class VariableCollection&lt;T, T2&gt; : GenericVariableCollectionBase</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Class VariableCollection(Of T, T2) _ Inherits GenericVariableCollectionBase</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>generic&lt;typename T, typename T2&gt; public ref class VariableCollection : public GenericVariableCollectionBase</code></td>
</tr>
</tbody>
</table>
Type Parameters

$T$
[Missing $<$typeparam name="T"$/> documentation for "T:Optimization.VariableCollection`2"]

$T2$
[Missing $<$typeparam name="T2"$/> documentation for "T:Optimization.VariableCollection`2"]
Inheritance Hierarchy

System.Object
Optimization.GenericVariableCollectionBase
Optimization.VariableCollection(T, T2)
See Also

VariableCollection(T, T2) Members
Optimization Namespace
The `VariableCollection(T, T2)` type exposes the following members.
### Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>VariableCollection(T, T2)(Func(T, T2, StringBuilder), Double, Double, VariableType, IEnumerable(T), IEnumerable(T2))</code></td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
<tr>
<td><code>VariableCollection(T, T2)(String, Double, Double, VariableType, IEnumerable(T), IEnumerable(T2))</code></td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable. (Inherited from <code>GenericVariableCollectionBase</code>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
</tbody>
</table>
### Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_internalVariableCollection</td>
<td>(Inherited from <a href="#">GenericVariableCollectionBase</a>)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index Validation</td>
<td>Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on (Inherited from <a href="#">GenericVariableCollectionBase</a>).</td>
</tr>
<tr>
<td>Item</td>
<td>Gets the <a href="#">Variable</a> with the specified index.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2) Class
Optimization Namespace
Optimization Framework

VariableCollection($T, T2$) Constructor

VariableCollection($T, T2$) Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VariableCollection</strong>(T, T2)(Func(T, T2, StringBuilder), Double, Double, VariableType, IEnumerable(T), IEnumerable(T2))</td>
<td>Initializes a new instance of the VariableCollection class.</td>
</tr>
<tr>
<td><strong>VariableCollection</strong>(T, T2)(String, Double, Double, VariableType, IEnumerable(T), IEnumerable(T2))</td>
<td>Initializes a new instance of the VariableCollection class.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection\((T, T2)\) Class
VariableCollection\((T, T2)\) Members
Optimization Namespace
Initializes a new instance of the `VariableCollection` class.

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
<tr>
<td>Public</td>
<td></td>
</tr>
<tr>
<td>VariableCollection(</td>
<td></td>
</tr>
<tr>
<td><code>Func&lt;T, T2, StringBuilder&gt;</code> uniqueNamegenerator</td>
<td></td>
</tr>
<tr>
<td><code>double</code> lowerBound,</td>
<td></td>
</tr>
<tr>
<td><code>double</code> upperBound,</td>
<td></td>
</tr>
<tr>
<td><code>VariableType</code> type,</td>
<td></td>
</tr>
<tr>
<td><code>IEnumerable&lt;T&gt;</code> set1,</td>
<td></td>
</tr>
<tr>
<td><code>IEnumerable&lt;T2&gt;</code> set2</td>
<td></td>
</tr>
<tr>
<td>)</td>
<td></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td></td>
</tr>
<tr>
<td>Public Sub New (</td>
<td></td>
</tr>
<tr>
<td><code>uniqueNamegenerator As Func(Of T, T2, StringBuilder)</code></td>
<td></td>
</tr>
<tr>
<td><code>lowerBound As Double</code>,</td>
<td></td>
</tr>
<tr>
<td><code>upperBound As Double</code>,</td>
<td></td>
</tr>
<tr>
<td><code>type As VariableType</code>,</td>
<td></td>
</tr>
<tr>
<td><code>set1 As IEnumerable(Of T)</code>,</td>
<td></td>
</tr>
<tr>
<td><code>set2 As IEnumerable(Of T2)</code></td>
<td></td>
</tr>
<tr>
<td>)</td>
<td></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
</tr>
<tr>
<td>public:</td>
<td></td>
</tr>
<tr>
<td>VariableCollection(</td>
<td></td>
</tr>
<tr>
<td><code>Func&lt;T, T2, StringBuilder&gt;</code> uniqueNamegenerator</td>
<td></td>
</tr>
<tr>
<td><code>double</code> lowerBound,</td>
<td></td>
</tr>
<tr>
<td><code>double</code> upperBound,</td>
<td></td>
</tr>
<tr>
<td><code>VariableType</code> type,</td>
<td></td>
</tr>
<tr>
<td><code>IEnumerable&lt;T&gt;</code> set1,</td>
<td></td>
</tr>
<tr>
<td><code>IEnumerable&lt;T2&gt;</code> set2</td>
<td></td>
</tr>
<tr>
<td>)</td>
<td></td>
</tr>
</tbody>
</table>
**Parameters**

*uniqueNamegenerator*
Type: `System.Func<T, T2, StringBuilder>`
A unique namegenerator.

*lowerBound*
Type: `System.Double`
The lower bound.

*upperBound*
Type: `System.Double`
The upper bound.

*type*
Type: `Optimization.VariableType`
The type.

*set1*
Type: `System.Collections.Generic.IEnumerable<T>`
The first set

*set2*
Type: `System.Collections.Generic.IEnumerable<T2>`
The second set
See Also

VariableCollection(T, T2) Class
VariableCollection(T, T2) Overload
Optimization Namespace
Initializes a new instance of the `VariableCollection` class.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
</tr>
</thead>
<tbody>
<tr>
<td>public VariableCollection(</td>
</tr>
<tr>
<td>string name,</td>
</tr>
<tr>
<td>double lowerBound,</td>
</tr>
<tr>
<td>double upperBound,</td>
</tr>
<tr>
<td>VariableType type,</td>
</tr>
<tr>
<td>IEnumerable&lt;T&gt; set1,</td>
</tr>
<tr>
<td>IEnumerable&lt;T2&gt; set2</td>
</tr>
<tr>
<td>)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual Basic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Sub New (</td>
</tr>
<tr>
<td>name As String,</td>
</tr>
<tr>
<td>lowerBound As Double,</td>
</tr>
<tr>
<td>upperBound As Double,</td>
</tr>
<tr>
<td>type As VariableType,</td>
</tr>
<tr>
<td>set1 As IEnumerable(Of T),</td>
</tr>
<tr>
<td>set2 As IEnumerable(Of T2)</td>
</tr>
<tr>
<td>)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>public: VariableCollection(</td>
</tr>
<tr>
<td>String^ name,</td>
</tr>
<tr>
<td>double lowerBound,</td>
</tr>
<tr>
<td>double upperBound,</td>
</tr>
<tr>
<td>VariableType type,</td>
</tr>
<tr>
<td>IEnumerable&lt;T&gt;^ set1,</td>
</tr>
<tr>
<td>IEnumerable&lt;T2&gt;^ set2</td>
</tr>
<tr>
<td>)</td>
</tr>
</tbody>
</table>
**Parameters**

*name*
Type: System.String
The name.

*lowerBound*
Type: System.Double
The lower bound.

*upperBound*
Type: System.Double
The upper bound.

*type*
Type: Optimization.VariableType
The type.

*set1*
Type: System.Collections.Generic.IEnumerable(T)
The first set

*set2*
Type: System.Collections.Generic.IEnumerable(T2)
The second set
See Also

VariableCollection(T, T2) Class
VariableCollection(T, T2) Overload
Optimization Namespace
The `VariableCollection(T, T2)` type exposes the following members.
### Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>internalVariableCollection</code></td>
<td>(Inherited from <a href="#">GenericVariableCollectionBase</a>)</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2) Class
Optimization Namespace
VariableCollection($T$, $T2$) Methods

The `VariableCollection(T, T2)` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is equal to the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.type">Type</a> of the current instance. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.collections.generic.genericvariablecollectionbase">GenericVariableCollectionBase</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="https://docs.microsoft.com/en-us/dotnet/api/system.string">String</a> that represents the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2) Class
Optimization Namespace
The `VariableCollection(T, T2)` type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Index Validation</strong></td>
<td>Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on (Inherited from <a href="#">GenericVariableCollectionBase</a>).</td>
</tr>
<tr>
<td><strong>Item</strong></td>
<td>Gets the <a href="#">Variable</a> with the specified index.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2) Class
Optimization Namespace
Gets the Variable with the specified index.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public Variable this[
    T element1,
    T2 element2
] { get; }
```

#### Visual Basic

```vbnet
Public ReadOnly Default Property Item ( _
    element1 As T,
    element2 As T2 _
) As Variable
    Get
```

#### Visual C++

```cpp
public:
property Variable^ default[T element1, T2 element2] 
    Variable^ get (T element1, T2 element2);
```

### Parameters

- **element1**
  - Type: **T**

- **element2**
  - Type: **T2**
See Also

VariableCollection(T, T2) Class
Optimization Namespace
VariableCollection\((T, T2, T3)\) Class

This is a strongly typed helper class that allows you to model your Variables more intuitively.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public class VariableCollection&lt;T, T2, T3&gt; : Generic</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Class VariableCollection(Of T, T2, T3) _ Inherits GenericVariableCollectionBase</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>generic&lt;typename T, typename T2, typename T3&gt; public ref class VariableCollection : public Generic</code></td>
</tr>
</tbody>
</table>
Type Parameters

T
[Missing <typeparam name="T"/> documentation for "T:Optimization.VariableCollection`3"]

T2
[Missing <typeparam name="T2"/> documentation for "T:Optimization.VariableCollection`3"]

T3
[Missing <typeparam name="T3"/> documentation for "T:Optimization.VariableCollection`3"]
Inheritance Hierarchy

System.Object
Optimization.GenericVariableCollectionBase
Optimization.VariableCollection(T, T2, T3)
See Also

- VariableCollection(T, T2, T3) Members
- Optimization Namespace
The *VariableCollection*(\(T, T2, T3\)) type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>VariableCollection&lt;T, T2, T3&gt;(Func&lt;T, T2, T3, StringBuilder&gt;, Double, Double, VariableType, IEnumerable&lt;T&gt;, IEnumerable&lt;T2&gt;, IEnumerable&lt;T3&gt;)</code></td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
<tr>
<td><code>VariableCollection&lt;T, T2, T3&gt;(String, Double, Double, VariableType, IEnumerable&lt;T&gt;, IEnumerable&lt;T2&gt;, IEnumerable&lt;T3&gt;)</code></td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
</tbody>
</table>
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable. (Inherited from <a href="#">GenericVariableCollectionBase</a>)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
</tbody>
</table>
### Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_internalVariableCollection</td>
<td>(Inherited from GenericVariableCollectionBase.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="index-validation" /></td>
<td><strong>Index Validation</strong>&lt;br&gt;Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on (Inherited from <a href="#">GenericVariableCollectionBase</a>).</td>
</tr>
<tr>
<td><img src="image" alt="item" /></td>
<td><strong>Item</strong>&lt;br&gt;Gets the <a href="#">Variable</a> with the specified index.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3) Class
Optimization Namespace
Optimization Framework

VariableCollection($T, T2, T3$) Constructor

VariableCollection($T, T2, T3$) Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>VariableCollection&lt;T, T2, T3&gt;(Func&lt;T, T2, T3, StringBuilder), Double, Double, VariableType, IEnumerable&lt;T&gt;, IEnumerable&lt;T2&gt;, IEnumerable&lt;T3&gt;)</code></td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
<tr>
<td><code>VariableCollection&lt;T, T2, T3&gt;(String, Double, Double, VariableType, IEnumerable&lt;T&gt;, IEnumerable&lt;T2&gt;, IEnumerable&lt;T3&gt;)</code></td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3) Class
VariableCollection(T, T2, T3) Members
Optimization Namespace
Initializes a new instance of the **VariableCollection** class.

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
</tr>
</thead>
</table>
| public VariableCollection(
  *Func*<T, T2, T3, *StringBuilder*> _uniqueNamegenerator_,
  *double* _lowerBound_,
  *double* _upperBound_,
  *VariableType* _type_,
  *IEnumerable*<T> _set1_,
  *IEnumerable*<T2> _set2_,
  *IEnumerable*<T3> _set3_)
| |

<table>
<thead>
<tr>
<th>Visual Basic</th>
</tr>
</thead>
</table>
| Public Sub New (_
  _uniqueNamegenerator As *Func*(Of T, T2, T3, *StringBuilder*), _
  _lowerBound As *Double*, _
  _upperBound As *Double*, _
  _type As *VariableType*, _
  _set1 As *IEnumerable*(Of T), _
  _set2 As *IEnumerable*(Of T2), _
  _set3 As *IEnumerable*(Of T3) _
| |

<table>
<thead>
<tr>
<th>Visual C++</th>
</tr>
</thead>
</table>
| public:
  VariableCollection(
    *Func*<T, T2, T3, *StringBuilder*>^>^ _uniqueNamegenerator_,
    *double* _lowerBound_,
    *double* _upperBound_,
    *VariableType* _type_,
    *IEnumerable*<T>^>^ _set1_,
    *IEnumerable*<T2>^>^ _set2_,
| |
Parameters

uniqueNamegenerator
Type: System.Func(T, T2, T3, StringBuilder)
A unique namegenerator.

lowerBound
Type: System.Double
The lower bound.

upperBound
Type: System.Double
The upper bound.

type
Type: Optimization.VariableType
The type.

set1
Type: System.Collections.Generic.IEnumerable(T)
The first set

set2
Type: System.Collections.Generic.IEnumerable(T2)
The second set

set3
Type: System.Collections.Generic.IEnumerable(T3)
The third set
See Also

VariableCollection(T, T2, T3) Class
VariableCollection(T, T2, T3) Overload
Optimization Namespace
Initializes a new instance of the `VariableCollection` class.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

**C#**

```csharp
public VariableCollection(
    string name,
    double lowerBound,
    double upperBound,
    VariableType type,
    IEnumerable<T> set1,
    IEnumerable<T2> set2,
    IEnumerable<T3> set3
)
```

**Visual Basic**

```vbnet
Public Sub New (_
    name As String, _
    lowerBound As Double, _
    upperBound As Double, _
    type As VariableType, _
    set1 As IEnumerable(Of T), _
    set2 As IEnumerable(Of T2), _
    set3 As IEnumerable(Of T3) _
)
```

**Visual C++**

```cpp
public:
VariableCollection(
    String^ name,
    double lowerBound,
    double upperBound,
    VariableType type,
    IEnumerable<T>^ set1,
    IEnumerable<T2>^ set2,
```
Parameters

name
Type: System.String
The name.

lowerBound
Type: System.Double
The lower bound.

upperBound
Type: System.Double
The upper bound.

type
Type: Optimization.VariableType
The type.

set1
Type: System.Collections.Generic.IEnumerable<T>
The first set

set2
Type: System.Collections.Generic.IEnumerable<T2>
The second set

set3
Type: System.Collections.Generic.IEnumerable<T3>
The third set
See Also

VariableCollection(T, T2, T3) Class
VariableCollection(T, T2, T3) Overload
Optimization Namespace
The `VariableCollection(T, T2, T3)` type exposes the following members.
## Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_internalVariableCollection</td>
<td>(Inherited from <a href="#">GenericVariableCollectionBase</a>)</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3) Class
Optimization Namespace
Optimization Framework

VariableCollection($T, T_2, T_3$) Methods

The VariableCollection($T, T_2, T_3$) type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this <a href="#">VariableCollection</a> to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable. (Inherited from <a href="#">GenericVariableCollectionBase</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3) Class
Optimization Namespace
Optimization Framework

VariableCollection($T$, $T2$, $T3$) Properties

The `VariableCollection($T$, $T2$, $T3$)` type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Icon] Index Validation</td>
<td>Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on (Inherited from <a href="#">GenericVariableCollectionBase</a>).</td>
</tr>
<tr>
<td>![Icon] Item</td>
<td>Gets the <a href="#">Variable</a> with the specified index.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3) Class
Optimization Namespace
Gets the `Variable` with the specified index.

**Namespace:** [Optimization](https://example.com)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
</table>
| `public Variable this[  
  T element1,  
  T2 element2,  
  T3 element3  
] { get; }` | `Public ReadOnly Default Property Item ( _  
  element1 As T, _  
  element2 As T2, _  
  element3 As T3 _  
) As Variable  
  Get` | `public:  
  property Variable^ default[T element1, T2 element2, _  
  Variable^ get (T element1, T2 element2, T3 e. ]` |

**Parameters**

- `element1`  
  Type: `T`  
- `element2`  
  Type: `T2`  
- `element3`  
  Type: `T3`
See Also

VariableCollection(T, T2, T3) Class
Optimization Namespace
This is a strongly typed helper class that allows you to model your Variables more intuitively

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public class VariableCollection&lt;T, T2, T3, T4&gt; : GenericVariableCollectionBase</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Class VariableCollection(Of T, T2, T3, T4) inherits GenericVariableCollectionBase</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>generic&lt;typename T, typename T2, typename T3, typename T4&gt; public ref class VariableCollection : public GenericVariableCollectionBase</code></td>
</tr>
</tbody>
</table>
### Type Parameters

*T*

[Missing `<typeparam name="T"/> documentation for "T:Optimization.VariableCollection`4"]

*T2*

[Missing `<typeparam name="T2"/> documentation for "T:Optimization.VariableCollection`4"]

*T3*

[Missing `<typeparam name="T3"/> documentation for "T:Optimization.VariableCollection`4"]

*T4*

[Missing `<typeparam name="T4"/> documentation for "T:Optimization.VariableCollection`4"]
Inheritance Hierarchy

System.Object

Optimization, GenericVariableCollectionBase

Optimization.VariableCollection(T, T2, T3, T4)
See Also

VariableCollection(T, T2, T3, T4) Members
Optimization Namespace
The `VariableCollection(T, T2, T3, T4)` type exposes the following members.
### Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>VariableCollection&lt;T, T2, T3, T4&gt;(Func&lt;T, T2, T3, T4, StringBuilder), Double, Double, VariableType, IEnumerable&lt;T&gt;, IEnumerable&lt;T2&gt;, IEnumerable&lt;T3&gt;, IEnumerable&lt;T4&gt;)</code></td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
<tr>
<td><code>VariableCollection&lt;T, T2, T3, T4&gt;(String, Double, Double, VariableType, IEnumerable&lt;T&gt;, IEnumerable&lt;T2&gt;, IEnumerable&lt;T3&gt;, IEnumerable&lt;T4&gt;)</code></td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
</tbody>
</table>
# Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code> (Inherited from <code>Object</code>).</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable. (Inherited from <code>GenericVariableCollectionBase</code>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
</tbody>
</table>
### Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_internalVariableCollection</td>
<td>(Inherited from GenericVariableCollectionBase.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index Validation</td>
<td>Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on (Inherited from GenericVariableCollectionBase.)</td>
</tr>
<tr>
<td>Item</td>
<td>Gets the Variable with the specified index.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4) Class
Optimization Namespace
VariableCollection\((T, T2, T3, T4)\) Constructor
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VariableCollection(T, T2, T3, T4)(Func(T, T2, T3, T4, StringBuilder), Double, Double, VariableType, IEnumerable(T), IEnumerable(T2), IEnumerable(T3), IEnumerable(T4))</td>
<td>Initializes a new instance of the VariableCollection class.</td>
</tr>
<tr>
<td>VariableCollection(T, T2, T3, T4)(String, Double, Double, VariableType, IEnumerable(T), IEnumerable(T2), IEnumerable(T3), IEnumerable(T4))</td>
<td>Initializes a new instance of the VariableCollection class.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection\(T, T_2, T_3, T_4\) Class
VariableCollection\(T, T_2, T_3, T_4\) Members
Optimization Namespace
VariableCollection\(T, T2, T3, T4\) Constructor (Func\(T, T2, T3, T4,\) StringBuilder), Double, Double, VariableType, IEnumerable\(T\), IEnumerable\(T2\), IEnumerable\(T3\), IEnumerable\(T4\))

Initializes a new instance of the VariableCollection class.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
</table>
| public VariableCollection(  
  Func<T, T2, T3, T4, StringBuilder> uniqueNamegenerator  
  double lowerBound,  
  double upperBound,  
  VariableType type,  
  IEnumerable<T> set1,  
  IEnumerable<T2> set2,  
  IEnumerable<T3> set3,  
  IEnumerable<T4> set4  
) | Public Sub New (  
  uniqueNamegenerator As Func(Of T, T2, T3, T4,  
  lowerBound As Double, _  
  upperBound As Double, _  
  type As VariableType, _  
  set1 As IEnumerable(Of T), _  
  set2 As IEnumerable(Of T2), _  
  set3 As IEnumerable(Of T3), _  
  set4 As IEnumerable(Of T4) _  
) | public:  
VariableCollection(  
  Func<T, T2, T3, T4, StringBuilder>^>^ uniqueNamegenerator  
  double lowerBound,  
  double upperBound,  
  VariableType type,  
)
```csharp
IEEnumerable<T>^ set1,
IEEnumerable<T2>^ set2,
IEEnumerable<T3>^ set3,
IEEnumerable<T4>^ set4
)
```

**Parameters**

`uniqueNamegenerator`
Type: `System.Func<T, T2, T3, T4, StringBuilder>`
A unique name generator.

`lowerBound`
Type: `System.Double`
The lower bound.

`upperBound`
Type: `System.Double`
The upper bound.

`type`
Type: `Optimization.VariableType`
The type.

`set1`
Type: `System.Collections.Generic.IEnumerable<T>`
The first set

`set2`
Type: `System.Collections.Generic.IEnumerable<T2>`
The second set

`set3`
Type: `System.Collections.Generic.IEnumerable<T3>`
The third set

`set4`
Type: `System.Collections.Generic.IEnumerable<T4>`
The fourth set
See Also

VariableCollection(T, T2, T3, T4) Class
VariableCollection(T, T2, T3, T4) Overload
Optimization Namespace
Initializes a new instance of the `VariableCollection` class.

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
</table>
| **C#**   | ```
public VariableCollection(
    string name,
    double lowerBound,
    double upperBound,
    VariableType type,
    IEnumerable<T> set1,
    IEnumerable<T2> set2,
    IEnumerable<T3> set3,
    IEnumerable<T4> set4
)
``` |
| **Visual Basic** | ```
Public Sub New (_
    name As String, _
    lowerBound As Double, _
    upperBound As Double, _
    type As VariableType, _
    set1 As IEnumerable(Of T), _
    set2 As IEnumerable(Of T2), _
    set3 As IEnumerable(Of T3), _
    set4 As IEnumerable(Of T4) _
)
``` |
| **Visual C++** | ```
public:
VariableCollection(
    String^ name,
    double lowerBound,
    double upperBound,
    VariableType type,
``` |
**Parameters**

*name*
Type: `System.String`
The name.

*lowerBound*
Type: `System.Double`
The lower bound.

*upperBound*
Type: `System.Double`
The upper bound.

*type*
Type: `Optimization.VariableType`
The type.

*set1*
Type: `System.Collections.Generic.IEnumerable<T>`
The first set

*set2*
Type: `System.Collections.Generic.IEnumerable<T2>`
The second set

*set3*
Type: `System.Collections.Generic.IEnumerable<T3>`
The third set

*set4*
Type: `System.Collections.Generic.IEnumerable<T4>`
The fourth set
### See Also

- [VariableCollection(T, T2, T3, T4) Class](#)
- [VariableCollection(T, T2, T3, T4) Overload](#)
- [Optimization Namespace](#)
The `VariableCollection(T, T2, T3, T4)` type exposes the following members.
## Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>_internalVariableCollection</code></td>
<td>(Inherited from <a href="#">GenericVariableCollectionBase</a>.)</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4) Class
Optimization Namespace
The `VariableCollection(T, T2, T3, T4)` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is equal to the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.type">Type</a> of the current instance. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">GenericVariableCollectionBase</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="https://docs.microsoft.com/en-us/dotnet/api/system.string">String</a> that represents the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

VariableCollection\((T, T2, T3, T4)\) Class

Optimization Namespace
VariableCollection($T$, $T2$, $T3$, $T4$) Properties

The VariableCollection($T$, $T2$, $T3$, $T4$) type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index Validation</td>
<td>Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on (Inherited from <a href="#">GenericVariableCollectionBase</a>).</td>
</tr>
<tr>
<td>Item</td>
<td>Gets the <a href="#">Variable</a> with the specified index.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4) Class
Optimization Namespace
Gets the Variable with the specified index.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public Variable this[
    T element1,
    T2 element2,
    T3 element3,
    T4 element4
] { get; }
```

### Visual Basic

```vbnet
Public ReadOnly Default Property Item ( _
    element1 As T, _
    element2 As T2, _
    element3 As T3, _
    element4 As T4 _
) As Variable
    Get
```

### Visual C++

```cpp
public:
    property Variable^ default[T element1, T2 element2, _
        Variable^ get (T element1, T2 element2, T3 e.

```

### Parameters

- **element1**
  Type: `T`

- **element2**
  Type: `T2`

- **element3**
Type: T3

*element4*
Type: T4
See Also

VariableCollection(T, T2, T3, T4) Class
Optimization Namespace
This is a strongly typed helper class that allows you to model your Variables more intuitively.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td><code>public class VariableCollection&lt;T, T2, T3, T4, T5&gt; :</code></td>
</tr>
<tr>
<td>Visual Basic</td>
<td><code>Public Class VariableCollection(Of T, T2, T3, T4, T5) Inherits GenericVariableCollectionBase</code></td>
</tr>
</tbody>
</table>
| Visual C++    | `generic<typename T, typename T2, typename T3, typename T4, typename T5>
public ref class VariableCollection : public Generic` |
Type Parameters

\( T \)

[Missing <typeparam name="T"/> documentation for "T:Optimization.VariableCollection`5"]

\( T2 \)

[Missing <typeparam name="T2"/> documentation for "T:Optimization.VariableCollection`5"]

\( T3 \)

[Missing <typeparam name="T3"/> documentation for "T:Optimization.VariableCollection`5"]

\( T4 \)

[Missing <typeparam name="T4"/> documentation for "T:Optimization.VariableCollection`5"]

\( T5 \)

[Missing <typeparam name="T5"/> documentation for "T:Optimization.VariableCollection`5"]
**Inheritance Hierarchy**

`System.Object`

`Optimization.GenericVariableCollectionBase`

`Optimization.VariableCollection(T, T2, T3, T4, T5)`
See Also

VariableCollection(T, T2, T3, T4, T5) Members
Optimization Namespace
VariableCollection($T, T_2, T_3, T_4, T_5$) Members

The `VariableCollection(T, T_2, T_3, T_4, T_5)` type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VariableCollection(T, T2, T3, T4, T5, Func(T, T2, T3, T4, T5, StringBuilder), Double, Double, VariableType, IEnumerable(T), IEnumerable(T2), IEnumerable(T3), IEnumerable(T4), IEnumerable(T5))</td>
<td>Initializes a new instance of the VariableCollection class.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VariableCollection(T, T2, T3, T4, T5)(String, Double, Double, VariableType, IEnumerable(T), IEnumerable(T2), IEnumerable(T3), IEnumerable(T4), IEnumerable(T5))</td>
<td>Initializes a new instance of the VariableCollection class.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this <code>VariableCollection</code> to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable. (Inherited from <code>GenericVariableCollectionBase</code>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
</tbody>
</table>
### Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_internalVariableCollection</td>
<td>(Inherited from GenericVariableCollectionBase.)</td>
</tr>
</tbody>
</table>


## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Index Validation</strong></td>
<td>Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on (Inherited from <a href="#">GenericVariableCollectionBase</a>).</td>
</tr>
<tr>
<td><strong>Item</strong></td>
<td>Gets the <a href="#">Variable</a> with the specified index.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4, T5) Class
Optimization Namespace
Optimization Framework

**VariableCollection**(T, T2, T3, T4, T5) Constructor

[VariableCollection](#) [T, T2, T3, T4, T5] Class [See Also](#) [Send Feedback](#)
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>VariableCollection(T, T2, T3, T4, T5)(Func(T, T2, T3, T4, T5, StringBuilder), Double, Double, VariableType, IEnumerable(T), IEnumerable(T2), IEnumerable(T3), IEnumerable(T4), IEnumerable(T5))</code></td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
<tr>
<td><code>VariableCollection(T, T2, T3, T4, T5)(String, Double, Double, VariableType, IEnumerable(T), IEnumerable(T2), IEnumerable(T3), IEnumerable(T4), IEnumerable(T5))</code></td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4, T5) Class
VariableCollection(T, T2, T3, T4, T5) Members
Optimization Namespace
Initializes a new instance of the `VariableCollection` class.

**Namespace:** [Optimization](#)  
**Assembly:** Optimization/Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#** | public VariableCollection(  
  Func<T, T2, T3, T4, T5, StringBuilder> uniqueNamegenerator,  
  double lowerBound,  
  double upperBound,  
  VariableType type,  
  IEnumerable<T> set1,  
  IEnumerable<T2> set2,  
  IEnumerable<T3> set3,  
  IEnumerable<T4> set4,  
  IEnumerable<T5> set5  
) |
| **Visual Basic** | Public Sub New (  
  uniqueNamegenerator As Func(Of T, T2, T3, T4, T5),  
  lowerBound As Double, _  
  upperBound As Double, _  
  type As VariableType, _  
  set1 As IEnumerable(Of T), _  
  set2 As IEnumerable(Of T2), _  
  set3 As IEnumerable(Of T3), _  
  set4 As IEnumerable(Of T4), _  
  set5 As IEnumerable(Of T5) _  
) |
| **Visual C++** | public:  
VariableCollection(  
  Func<T, T2, T3, T4, T5, StringBuilder^>^ uniqueNamegenerator,  
  double lowerBound,  
  double upperBound,  
  VariableType type,  
  IEnumerable<T> set1,  
  IEnumerable<T2> set2,  
  IEnumerable<T3> set3,  
  IEnumerable<T4> set4,  
  IEnumerable<T5> set5  
) |
double upperBound,
VariableType type,
IEnumerable<T>^ set1,
IEnumerable<T2>^ set2,
IEnumerable<T3>^ set3,
IEnumerable<T4>^ set4,
IEnumerable<T5>^ set5

Parameters

uniqueNamegenerator
Type: System.Func(T, T2, T3, T4, T5, StringBuilder)
A unique namegenerator.

lowerBound
Type: System.Double
The lower bound.

upperBound
Type: System.Double
The upper bound.

type
Type: Optimization.VariableType
The type.

set1
Type: System.Collections.Generic(IEnumerable<T>)
The first set

set2
Type: System.Collections.Generic(IEnumerable<T2>)
The second set

set3
Type: System.Collections.Generic(IEnumerable<T3>)
The third set

set4
Type: System.Collections.Generic(IEnumerable<T4>)
The fourth set

\textit{set5}

Type: \texttt{System.Collections.Generic.IEnumerable(T5)}

The fifth set
See Also

VariableCollection(T, T2, T3, T4, T5) Class
VariableCollection(T, T2, T3, T4, T5) Overload
Optimization Namespace
Optimization Framework

VariableCollection\((T, T2, T3, T4, T5)\) Constructor (String, Double, Double, VariableType, IEnumerable\(T\), IEnumerable\(T2\), IEnumerable\(T3\), IEnumerable\(T4\), IEnumerable\(T5\))

Initializes a new instance of the VariableCollection class.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

**C#**

```csharp
public VariableCollection(
    string name,
    double lowerBound,
    double upperBound,
    VariableType type,
    IEnumerable<T> set1,
    IEnumerable<T2> set2,
    IEnumerable<T3> set3,
    IEnumerable<T4> set4,
    IEnumerable<T5> set5
)
```

**Visual Basic**

```vbnet
Public Sub New ( _
    name As String, _
    lowerBound As Double, _
    upperBound As Double, _
    type As VariableType, _
    set1 As IEnumerable(Of T), _
    set2 As IEnumerable(Of T2), _
    set3 As IEnumerable(Of T3), _
    set4 As IEnumerable(Of T4), _
    set5 As IEnumerable(Of T5) _
)
```

**Visual C++**

```cpp
public:
VariableCollection(
    const String^ name,
    double lowerBound,
```
double upperBound,
VariableType type,
IEnumerable<T>^ set1,
IEnumerable<T2>^ set2,
IEnumerable<T3>^ set3,
IEnumerable<T4>^ set4,
IEnumerable<T5>^ set5
)

Parameters

name
Type: System.String
The name.

lowerBound
Type: System.Double
The lower bound.

upperBound
Type: System.Double
The upper bound.

type
Type: Optimization.VariableType
The type.

set1
Type: System.Collections.Generic.IEnumerable<T>
The first set

set2
Type: System.Collections.Generic.IEnumerable<T2>
The second set

set3
Type: System.Collections.Generic.IEnumerable<T3>
The third set

set4
Type: System.Collections.Generic.IEnumerable<T4>
The fourth set

set5
Type: System.Collections.Generic.IEnumerable(T5)

The fifth set
See Also

VariableCollection(T, T2, T3, T4, T5) Class

VariableCollection(T, T2, T3, T4, T5) Overload Optimization Namespace
VariableCollection($T, T2, T3, T4, T5$) Fields

The VariableCollection($T, T2, T3, T4, T5$) type exposes the following members.
### Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_internalVariableCollection</td>
<td>(Inherited from <a href="#">GenericVariableCollectionBase</a>)</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4, T5) Class
Optimization Namespace
Optimization Framework

**VariableCollection**(\(T, T_2, T_3, T_4, T_5\)) Methods

The **VariableCollection**(\(T, T_2, T_3, T_4, T_5\)) type exposes the following members.
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is equal to the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.type">Type</a> of the current instance. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.collections.specialized.genericvariablecollectionbase">GenericVariableCollectionBase</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="https://docs.microsoft.com/en-us/dotnet/api/system.string">String</a> that represents the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4, T5) Class
Optimization Namespace
The \texttt{VariableCollection}(T, T2, T3, T4, T5) type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index Validation</td>
<td>Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on (Inherited from GenericVariableCollectionBase.)</td>
</tr>
<tr>
<td>Item</td>
<td>Gets the Variable with the specified index.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4, T5) Class
Optimization Namespace
Gets the `Variable` with the specified index.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public Variable this[
    T element1,
    T2 element2,
    T3 element3,
    T4 element4,
    T5 element5
] { get; }
```

**Visual Basic**

```vbnet
Public ReadOnly Default Property Item ( _
    element1 As T, _
    element2 As T2, _
    element3 As T3, _
    element4 As T4, _
    element5 As T5 _
) As Variable
    Get
```

**Visual C++**

```cpp
public:
    property Variable^ default[T element1, T2 element2, T3 element3, T4 element4, T5 element5] { Variable^ get (T element1, T2 element2, T3 e. }
```

### Parameters

- **element1**
  - Type: **T**

- **element2**
Type: $T_2$

$element_3$
Type: $T_3$

$element_4$
Type: $T_4$

$element_5$
Type: $T_5$
See Also

- VariableCollection\((T, T2, T3, T4, T5)\) Class
- Optimization Namespace
VariableCollection\((T, T2, T3, T4, T5, T6)\) Class

This is a strongly typed helper class that allows you to model your Variables more intuitively.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td><code>public class VariableCollection&lt;T, T2, T3, T4, T5, T6&gt;</code></td>
</tr>
<tr>
<td>Visual Basic</td>
<td><code>Public Class VariableCollection(Of T, T2, T3, T4, T5, T6) Inherits GenericVariableCollectionBase</code></td>
</tr>
<tr>
<td>Visual C++</td>
<td><code>generic&lt;typename T, typename T2, typename T3, typename T4, typename T5, typename T6&gt; public ref class VariableCollection : public GenericVariableCollectionBase</code></td>
</tr>
</tbody>
</table>
Type Parameters

$T$
[Missing <typeparam name="T"/> documentation for "T:Optimization.VariableCollection`6"]

$T_2$
[Missing <typeparam name="T2"/> documentation for "T:Optimization.VariableCollection`6"]

$T_3$
[Missing <typeparam name="T3"/> documentation for "T:Optimization.VariableCollection`6"]

$T_4$
[Missing <typeparam name="T4"/> documentation for "T:Optimization.VariableCollection`6"]

$T_5$
[Missing <typeparam name="T5"/> documentation for "T:Optimization.VariableCollection`6"]

$T_6$
[Missing <typeparam name="T6"/> documentation for "T:Optimization.VariableCollection`6"]
Inheritance Hierarchy

System.Object
  Optimization.GenericVariableCollectionBase
    Optimization.VariableCollection(T, T2, T3, T4, T5, T6)
See Also

VariableCollection(T, T2, T3, T4, T5, T6) Members
Optimization Namespace
The **VariableCollection**(T, T₂, T₃, T₄, T₅, T₆) type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VariableCollection(T, T2, T3, T4, T5, T6) (Func(T, T2, T3, T4, T5, T6, StringBuilder), Double, Double, VariableType, IEnumerable(T), IEnumerable(T2), IEnumerable(T3), IEnumerable(T4), IEnumerable(T5), IEnumerable(T6))</td>
<td>Initializes a new instance of the VariableCollection class.</td>
</tr>
<tr>
<td>VariableCollection(T, T2, T3, T4, T5, T6) (String, Double, Double, VariableType, IEnumerable(T), IEnumerable(T2), IEnumerable(T3), IEnumerable(T4), IEnumerable(T5), IEnumerable(T6))</td>
<td>Initializes a new instance of the VariableCollection class.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable. (Inherited from <code>GenericVariableCollectionBase</code>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
</tbody>
</table>
## Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_internalVariableCollection</td>
<td>(Inherited from <a href="#">GenericVariableCollectionBase</a>)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>![icon] Index Validation</td>
<td>Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on (Inherited from <a href="#">GenericVariableCollectionBase</a>).</td>
</tr>
<tr>
<td>![icon] Item</td>
<td>Gets the <a href="#">Variable</a> with the specified index.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4, T5, T6) Class
Optimization Namespace
Optimization Framework

VariableCollection\((T, T2, T3, T4, T5, T6)\) Constructor

VariableCollection\((T, T2, T3, T4, T5, T6)\) Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VariableCollection(T, T2, T3, T4, T5, T6)</strong> (Func(T, T2, T3, T4, T5, T6, StringBuilder), Double, Double, VariableType, IEnumerable(T), IEnumerable(T2), IEnumerable(T3), IEnumerable(T4), IEnumerable(T5), IEnumerable(T6))</td>
<td>Initializes a new instance of the VariableCollection class.</td>
</tr>
<tr>
<td><strong>VariableCollection(T, T2, T3, T4, T5, T6)</strong> (String, Double, Double, VariableType, IEnumerable(T), IEnumerable(T2), IEnumerable(T3), IEnumerable(T4), IEnumerable(T5), IEnumerable(T6))</td>
<td>Initializes a new instance of the VariableCollection class.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4, T5, T6) Class
VariableCollection(T, T2, T3, T4, T5, T6) Members
Optimization Namespace
Initializes a new instance of the `VariableCollection` class.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
C#

```csharp
public VariableCollection(
    Func<T, T2, T3, T4, T5, T6, StringBuilder> uniqueNamegenerator,
    double lowerBound,
    double upperBound,
    VariableType type,
    IEnumerable<T> set1,
    IEnumerable<T2> set2,
    IEnumerable<T3> set3,
    IEnumerable<T4> set4,
    IEnumerable<T5> set5,
    IEnumerable<T6> set6
)
```

Visual Basic

```vbnet
Public Sub New ( _
    uniqueNamegenerator As Func(Of T, T2, T3, T4, 
    lowerBound As Double, _
    upperBound As Double, _
    type As VariableType, _
    set1 As IEnumerable(Of T), _
    set2 As IEnumerable(Of T2), _
    set3 As IEnumerable(Of T3), _
    set4 As IEnumerable(Of T4), _
    set5 As IEnumerable(Of T5), _
    set6 As IEnumerable(Of T6) _
)
```

Visual C++

```cpp
public:
VariableCollection(
```
Parameters

uniqueNamegenerator
Type: `System.Func<T, T2, T3, T4, T5, T6, StringBuilder>`
A unique namegenerator.

lowerBound
Type: `System.Double`
The lower bound.

upperBound
Type: `System.Double`
The upper bound.

type
Type: `Optimization.VariableType`
The type.

set1
Type: `System.Collections.Generic(IEnumerable<T>)`
The first set

set2
Type: `System.Collections.Generic(IEnumerable<T2>)`
The second set

set3
Type: `System.Collections.Generic(IEnumerable<T3>)`
The third set
*set4*
Type: `System.Collections.Generic.IEnumerable<T4>`
The fourth set

*set5*
Type: `System.Collections.Generic.IEnumerable<T5>`
The fifth set

*set6*
Type: `System.Collections.Generic.IEnumerable<T6>`
The sixth set
See Also

VariableCollection(T, T2, T3, T4, T5, T6) Class
VariableCollection(T, T2, T3, T4, T5, T6) Overload
Optimization Namespace
VariableCollection\((T, T2, T3, T4, T5, T6)\) Constructor (String, Double, Double, VariableType, IEnumerable\((T)\), IEnumerable\((T2)\), IEnumerable\((T3)\), IEnumerable\((T4)\), IEnumerable\((T5)\), IEnumerable\((T6)\))

Initializes a new instance of the VariableCollection class.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public VariableCollection(
    string name,
    double lowerBound,
    double upperBound,
    VariableType type,
    IEnumerable<T> set1,
    IEnumerable<T2> set2,
    IEnumerable<T3> set3,
    IEnumerable<T4> set4,
    IEnumerable<T5> set5,
    IEnumerable<T6> set6
)
```

**Visual Basic**

```vbnet
Public Sub New ( _
    name As String, _
    lowerBound As Double, _
    upperBound As Double, _
    type As VariableType, _
    set1 As IEnumerable(Of T), _
    set2 As IEnumerable(Of T2), _
    set3 As IEnumerable(Of T3), _
    set4 As IEnumerable(Of T4), _
    set5 As IEnumerable(Of T5), _
    set6 As IEnumerable(Of T6) _
)
```

**Visual C++**

```c++
public:
VariableCollection(
```
Parameters

name
Type: System.String
The name.

lowerBound
Type: System.Double
The lower bound.

upperBound
Type: System.Double
The upper bound.

type
Type: Optimization.VariableType
The type.

set1
Type: System.Collections.Generic.IEnumerable<T>
The first set

set2
Type: System.Collections.Generic.IEnumerable<T2>
The second set

set3
Type: System.Collections.Generic.IEnumerable<T3>
The third set
set4
Type: System.Collections.Generic.IEnumerable(T4)
The fourth set

set5
Type: System.Collections.Generic.IEnumerable(T5)
The fifth set

set6
Type: System.Collections.Generic.IEnumerable(T6)
The sixth set
See Also

VariableCollection(T, T2, T3, T4, T5, T6) Class
VariableCollection(T, T2, T3, T4, T5, T6) Overload
Optimization Namespace
The `VariableCollection(T, T2, T3, T4, T5, T6)` type exposes the following members.
### Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>_internalVariableCollection</code></td>
<td>(Inherited from <a href="#">GenericVariableCollectionBase</a>.)</td>
</tr>
</tbody>
</table>
See Also

VariableCollection\((T, T2, T3, T4, T5, T6)\) Class

Optimization Namespace
The `VariableCollection(T, T2, T3, T4, T5, T6)` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable. (Inherited from <a href="#">GenericVariableCollectionBase</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4, T5, T6) Class
Optimization Namespace
The \texttt{VariableCollection(T, T2, T3, T4, T5, T6)} type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index Validation</td>
<td>Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on (Inherited from <a href="#">GenericVariableCollectionBase</a>).</td>
</tr>
<tr>
<td>Item</td>
<td>Gets the <a href="#">Variable</a> with the specified index.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4, T5, T6) Class
Optimization Namespace
Gets the **Variable** with the specified index.

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

**C#**

```csharp
public Variable this[
    T element1,
    T2 element2,
    T3 element3,
    T4 element4,
    T5 element5,
    T6 element6
] { get; }
```

**Visual Basic**

```vbnet
Public ReadOnly Default Property Item ( _
    element1 As T, _
    element2 As T2, _
    element3 As T3, _
    element4 As T4, _
    element5 As T5, _
    element6 As T6 _
) As Variable
    Get
```

**Visual C++**

```cpp
public:
    property Variable^ default[T element1, T2 element2, T3 element3, T4 element4, T5 element5, T6 element6] { Variable^ get (T element1, T2 element2, T3 element3, T4 element4, T5 element5, T6 element6); }
```

### Parameters

- `element1`
Type: T

element2
Type: T2

element3
Type: T3

element4
Type: T4

element5
Type: T5

element6
Type: T6
See Also

VariableCollection(T, T2, T3, T4, T5, T6) Class
Optimization Namespace
This is a strongly typed helper class that allows you to model your Variables more intuitively.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th><strong>Syntax</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
</tr>
<tr>
<td>public class VariableCollection&lt;T, T2, T3, T4, T5, T6, T7&gt;</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
</tr>
<tr>
<td>Public Class VariableCollection(Of T, T2, T3, T4, T5, T6, T7)</td>
</tr>
<tr>
<td>Inherits GenericVariableCollectionBase</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
</tr>
<tr>
<td>generic&lt;typename T, typename T2, typename T3, typename T4, typename T5, typename T6, typename T7&gt;</td>
</tr>
<tr>
<td>public ref class VariableCollection : public GenericVariableCollectionBase</td>
</tr>
</tbody>
</table>
## Type Parameters

*T*

[Missing `<typeparam name="T"/>` documentation for "T:Optimization.VariableCollection`7"]

*T2*

[Missing `<typeparam name="T2"/>` documentation for "T:Optimization.VariableCollection`7"]

*T3*

[Missing `<typeparam name="T3"/>` documentation for "T:Optimization.VariableCollection`7"]

*T4*

[Missing `<typeparam name="T4"/>` documentation for "T:Optimization.VariableCollection`7"]

*T5*

[Missing `<typeparam name="T5"/>` documentation for "T:Optimization.VariableCollection`7"]

*T6*

[Missing `<typeparam name="T6"/>` documentation for "T:Optimization.VariableCollection`7"]

*T7*

[Missing `<typeparam name="T7"/>` documentation for "T:Optimization.VariableCollection`7"]
Inheritance Hierarchy

`System.Object`
`Optimization.GenericVariableCollectionBase`
`Optimization.VariableCollection(T, T2, T3, T4, T5, T6, T7)`
See Also

VariableCollection(T, T2, T3, T4, T5, T6, T7) Members
Optimization Namespace
The `VariableCollection(T, T2, T3, T4, T5, T6, T7)` type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>VariableCollection(T, T2, T3, T4, T5, T6, T7)</code> <em>(Func(T, T2, T3, T4, T5, T6, T7, StringBuilder), Double, Double, VariableType, IEnumerable(T), IEnumerable(T2), IEnumerable(T3), IEnumerable(T4), IEnumerable(T5), IEnumerable(T6), IEnumerable(T7)))</em></td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
<tr>
<td><code>VariableCollection(T, T2, T3, T4, T5, T6, T7)</code> <em>(String, Double, Double, VariableType, IEnumerable(T), IEnumerable(T2), IEnumerable(T3), IEnumerable(T4), IEnumerable(T5), IEnumerable(T6), IEnumerable(T7)))</em></td>
<td>Initializes a new instance of the <code>VariableCollection</code> class.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable. (Inherited from <code>GenericVariableCollectionBase</code>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
</tbody>
</table>
## Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>_internalVariableCollection</code></td>
<td>(Inherited from <a href="#">GenericVariableCollectionBase</a>.)</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>![icon]</td>
<td><strong>Index Validation</strong> Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on (Inherited from <a href="#">GenericVariableCollectionBase</a>.)</td>
</tr>
<tr>
<td>![icon]</td>
<td><strong>Item</strong> Gets the <a href="#">Variable</a> with the specified index.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4, T5, T6, T7) Class
Optimization Namespace
Optimization Framework

VariableCollection($T, T_2, T_3, T_4, T_5, T_6, T_7$) Constructor

VariableCollection($T, T_2, T_3, T_4, T_5, T_6, T_7$) Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VariableCollection(T, T2, T3, T4, T5, T6, T7) (Func(T, T2, T3, T4, T5, T6, T7, StringBuilder), Double, Double, VariableType, IEnumerable(T), IEnumerable(T2), IEnumerable(T3), IEnumerable(T4), IEnumerable(T5), IEnumerable(T6), IEnumerable(T7))</td>
<td>Initializes a new instance of the VariableCollection class.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VariableCollection(T, T2, T3, T4, T5, T6, T7) (String, Double, Double, VariableType, IEnumerable(T), IEnumerable(T2), IEnumerable(T3), IEnumerable(T4), IEnumerable(T5), IEnumerable(T6), IEnumerable(T7))</td>
<td>Initializes a new instance of the VariableCollection class.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4, T5, T6, T7) Class
VariableCollection(T, T2, T3, T4, T5, T6, T7) Members
Optimization Namespace
VariableCollection\((T, T2, T3, T4, T5, T6, T7)\) Constructor \((\text{Func}(T, T2, T3, T4, T5, T6, T7, \text{StringBuilder}), \text{Double}, \text{Double}, \text{VariableType}, \text{IEnumerable}(T), \text{IEnumerable}(T2), \text{IEnumerable}(T3), \text{IEnumerable}(T4), \text{IEnumerable}(T5), \text{IEnumerable}(T6), \text{IEnumerable}(T7))\)

Initializes a new instance of the \textbf{VariableCollection} class.

\textbf{Namespace:} Optimization
\textbf{Assembly:} Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public VariableCollection(
    Func<T, T2, T3, T4, T5, T6, T7, StringBuilder, double lowerBound,
    double upperBound,
    VariableType type,
    IEnumerable<T> set1,
    IEnumerable<T2> set2,
    IEnumerable<T3> set3,
    IEnumerable<T4> set4,
    IEnumerable<T5> set5,
    IEnumerable<T6> set6,
    IEnumerable<T7> set7
)
```

#### Visual Basic

```vbnet
Public Sub New (_
    uniqueNamegenerator As Func(Of T, T2, T3, T4, lowerBound As Double, _
    upperBound As Double, _
    type As VariableType, _
    set1 As IEnumerable(Of T), _
    set2 As IEnumerable(Of T2), _
    set3 As IEnumerable(Of T3), _
    set4 As IEnumerable(Of T4), _
    set5 As IEnumerable(Of T5), _
    set6 As IEnumerable(Of T6), _
    set7 As IEnumerable(Of T7)_
)
```

#### Visual C++
public:
VariableCollection(
    Func<T, T2, T3, T4, T5, T6, T7, StringBuilder, double lowerBound,
    double upperBound, VariableType type,
    IEnumerable<T>^ set1,
    IEnumerable<T2>^ set2,
    IEnumerable<T3>^ set3,
    IEnumerable<T4>^ set4,
    IEnumerable<T5>^ set5,
    IEnumerable<T6>^ set6,
    IEnumerable<T7>^ set7)
)

Parameters

uniqueNamegenerator
Type: System.Func(T, T2, T3, T4, T5, T6, T7, StringBuilder)
A unique namegenerator.

lowerBound
Type: System.Double
The lower bound.

upperBound
Type: System.Double
The upper bound.

type
Type: Optimization.VariableType
The type.

set1
Type: System.Collections.Generic.IEnumerable<T>
The first set

set2
Type: System.Collections.Generic.IEnumerable<T2>
The second set
set3
Type: System.Collections.Generic.IEnumerable<T3>
The third set

set4
Type: System.Collections.Generic.IEnumerable<T4>
The fourth set

set5
Type: System.Collections.Generic.IEnumerable<T5>
The fifth set

set6
Type: System.Collections.Generic.IEnumerable<T6>
The sixth set

set7
Type: System.Collections.Generic.IEnumerable<T7>
The seventh set
See Also

VariableCollection(T, T2, T3, T4, T5, T6, T7) Class
VariableCollection(T, T2, T3, T4, T5, T6, T7) Overload
Optimization Namespace
VariableCollection\((T, T2, T3, T4, T5, T6, T7)\) Constructor (String, Double, Double, VariableType, IEnumerable\((T)\), IEnumerable\((T2)\), IEnumerable\((T3)\), IEnumerable\((T4)\), IEnumerable\((T5)\), IEnumerable\((T6)\), IEnumerable\((T7)\))

Initializes a new instance of the **VariableCollection** class.

**Namespace:** Optimization  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
C#

```csharp
public VariableCollection(
    string name,
    double lowerBound,
    double upperBound,
    VariableType type,
    IEnumerable<T> set1,
    IEnumerable<T2> set2,
    IEnumerable<T3> set3,
    IEnumerable<T4> set4,
    IEnumerable<T5> set5,
    IEnumerable<T6> set6,
    IEnumerable<T7> set7
)
```

Visual Basic

```vbscript
Public Sub New (_
    name As String, _
    lowerBound As Double, _
    upperBound As Double, _
    type As VariableType, _
    set1 As IEnumerable(Of T), _
    set2 As IEnumerable(Of T2), _
    set3 As IEnumerable(Of T3), _
    set4 As IEnumerable(Of T4), _
    set5 As IEnumerable(Of T5), _
    set6 As IEnumerable(Of T6), _
    set7 As IEnumerable(Of T7) _
)
```

Visual C++
public:
VariableCollection(
    String^ name,
    double lowerBound,
    double upperBound,
    VariableType type,
    IEnumerable<T>^ set1,
    IEnumerable<T2>^ set2,
    IEnumerable<T3>^ set3,
    IEnumerable<T4>^ set4,
    IEnumerable<T5>^ set5,
    IEnumerable<T6>^ set6,
    IEnumerable<T7>^ set7
)

**Parameters**

*name*
Type: System.String
The name.

*lowerBound*
Type: System.Double
The lower bound.

*upperBound*
Type: System.Double
The upper bound.

*type*
Type: Optimization.VariableType
The type.

*set1*
Type: System.Collections.Generic.IEnumerable<T>
The first set

*set2*
Type: System.Collections.Generic.IEnumerable<T2>
The second set
set3
Type: `System.Collections.Generic(IEnumerable<T3>)`
The third set

set4
Type: `System.Collections.Generic(IEnumerable<T4>)`
The fourth set

set5
Type: `System.Collections.Generic(IEnumerable<T5>)`
The fifth set

set6
Type: `System.Collections.Generic(IEnumerable<T6>)`
The sixth set

set7
Type: `System.Collections.Generic(IEnumerable<T7>)`
The seventh set
See Also

VariableCollection(T, T2, T3, T4, T5, T6, T7) Class
VariableCollection(T, T2, T3, T4, T5, T6, T7) Overload
Optimization Namespace
The `VariableCollection(T, T2, T3, T4, T5, T6, T7)` type exposes the following members.
## Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>_internalVariableCollection</code></td>
<td>(Inherited from <a href="#">GenericVariableCollectionBase</a>.)</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4, T5, T6, T7) Class

Optimization Namespace
The `VariableCollection(T, T2, T3, T4, T5, T6, T7)` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is equal to the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.type">Type</a> of the current instance. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.genericvariablecollectionbase">GenericVariableCollectionBase</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="https://docs.microsoft.com/en-us/dotnet/api/system.string">String</a> that represents the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4, T5, T6, T7) Class
Optimization Namespace
The **VariableCollection**\( (T, T_2, T_3, T_4, T_5, T_6, T_7) \) type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index Validation</td>
<td>Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on (Inherited from GenericVariableCollectionBase.)</td>
</tr>
<tr>
<td>Item</td>
<td>Gets the Variable with the specified index.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4, T5, T6, T7) Class
Optimization Namespace
Gets the Variable with the specified index.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public Variable this[
    T element1,
    T2 element2,
    T3 element3,
    T4 element4,
    T5 element5,
    T6 element6,
    T7 element7
] { get; }
```

**Visual Basic**

```vbnet
Public ReadOnly Default Property Item ( _
    element1 As T, _
    element2 As T2, _
    element3 As T3, _
    element4 As T4, _
    element5 As T5, _
    element6 As T6, _
    element7 As T7 _
) As Variable
Get
```

**Visual C++**

```cpp
public:
    property Variable^ default[T element1, T2 element2, T3 element3, T4 element4, T5 element5, T6 element6, T7 element7]
    Variable^ get (T element1, T2 element2, T3 element3, T4 element4, T5 element5, T6 element6, T7 element7)
}
```

### Parameters
element1
Type: T

element2
Type: T2

element3
Type: T3

element4
Type: T4

element5
Type: T5

element6
Type: T6

element7
Type: T7
See Also

VariableCollection(T, T2, T3, T4, T5, T6, T7) Class
Optimization Namespace
VariableCollection\((T, T2, T3, T4, T5, T6, T7, T8)\) Class

This is a strongly typed helper class that allows you to model your Variables more intuitively.

**Namespace:** [Optimization](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public class VariableCollection&lt;T, T2, T3, T4, T5, T6, T7, T8&gt;</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Class VariableCollection(Of T, T2, T3, T4, T5, T6, T7, T8) Inherits GenericVariableCollectionBase</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>generic&lt;typename T, typename T2, typename T3, typename T4, typename T5, typename T6, typename T7, typename T8&gt; public ref class VariableCollection : public GenericVariableCollectionBase</td>
</tr>
</tbody>
</table>
### Type Parameters

**T**

[Missing <typeparam name="T"/> documentation for "T:Optimization.VariableCollection`8"]

**T2**

[Missing <typeparam name="T2"/> documentation for "T:Optimization.VariableCollection`8"]

**T3**

[Missing <typeparam name="T3"/> documentation for "T:Optimization.VariableCollection`8"]

**T4**

[Missing <typeparam name="T4"/> documentation for "T:Optimization.VariableCollection`8"]

**T5**

[Missing <typeparam name="T5"/> documentation for "T:Optimization.VariableCollection`8"]

**T6**

[Missing <typeparam name="T6"/> documentation for "T:Optimization.VariableCollection`8"]

**T7**

[Missing <typeparam name="T7"/> documentation for "T:Optimization.VariableCollection`8"]

**T8**

[Missing <typeparam name="T8"/> documentation for "T:Optimization.VariableCollection`8"]
Inheritance Hierarchy

System.Object
Optimization.GenericVariableCollectionBase
    Optimization.VariableCollection(T, T2, T3, T4, T5, T6, T7, T8)
See Also

VariableCollection(T, T2, T3, T4, T5, T6, T7, T8) Members
Optimization Namespace
The `VariableCollection(T, T2, T3, T4, T5, T6, T7, T8)` type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VariableCollection(T, T2, T3, T4, T5, T6, T7, T8)(Func(T, T2, T3, T4, T5, T6, T7, T8, StringBuilder), Double, Double, VariableType, IEnumerable(T), IEnumerable(T2), IEnumerable(T3), IEnumerable(T4), IEnumerable(T5), IEnumerable(T6), IEnumerable(T7), IEnumerable(T8))</td>
<td>Initializes a new instance of the VariableCollection class.</td>
</tr>
<tr>
<td>VariableCollection(T, T2, T3, T4, T5, T6, T7, T8)(String, Double, Double, VariableType, IEnumerable(T), IEnumerable(T2), IEnumerable(T3), IEnumerable(T4), IEnumerable(T5), IEnumerable(T6), IEnumerable(T7), IEnumerable(T8))</td>
<td>Initializes a new instance of the VariableCollection class.</td>
</tr>
</tbody>
</table>
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is equal to the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.type">Type</a> of the current instance. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this VariableCollection to the corresponding value in the parameter 'values'. The matching happens by string comparison of the name of the variable. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.collections.generic.genericvariablecollectionbase">GenericVariableCollectionBase</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="https://docs.microsoft.com/en-us/dotnet/api/system.string">String</a> that represents the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
</tbody>
</table>
### Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_internalVariableCollection</td>
<td>(Inherited from GenericVariableCollectionBase)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Index Validation</strong></td>
<td>Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on (Inherited from <a href="#">GenericVariableCollectionBase</a>).</td>
</tr>
<tr>
<td><strong>Item</strong></td>
<td>Gets the <a href="#">Variable</a> with the specified index.</td>
</tr>
</tbody>
</table>
See Also

`VariableCollection<T, T2, T3, T4, T5, T6, T7, T8> Class`  
`Optimization Namespace`
Optimization Framework

VariableCollection\((T, T2, T3, T4, T5, T6, T7, T8)\) Constructor

See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VariableCollection(T, T2, T3, T4, T5, T6, T7, T8)(Func(T, T2, T3, T4, T5, T6, T7, T8, StringBuilder), Double, Double, VariableType, IEnumerable(T), IEnumerable(T2), IEnumerable(T3), IEnumerable(T4), IEnumerable(T5), IEnumerable(T6), IEnumerable(T7), IEnumerable(T8))</strong></td>
<td>Initializes a new instance of the <strong>VariableCollection</strong> class.</td>
</tr>
<tr>
<td><strong>VariableCollection(T, T2, T3, T4, T5, T6, T7, T8)(String, Double, Double, VariableType, IEnumerable(T), IEnumerable(T2), IEnumerable(T3), IEnumerable(T4), IEnumerable(T5), IEnumerable(T6), IEnumerable(T7), IEnumerable(T8))</strong></td>
<td>Initializes a new instance of the <strong>VariableCollection</strong> class.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4, T5, T6, T7, T8) Class
VariableCollection(T, T2, T3, T4, T5, T6, T7, T8) Members
Optimization Namespace
VariableCollection\( (T, T2, T3, T4, T5, T6, T7, T8) \) Constructor (Func\( (T, T2, T3, T4, T5, T6, T7, T8, StringBuilder) \), Double, Double, VariableType, IEnumerable\( (T) \), IEnumerable\( (T2) \), IEnumerable\( (T3) \), IEnumerable\( (T4) \), IEnumerable\( (T5) \), IEnumerable\( (T6) \), IEnumerable\( (T7) \), IEnumerable\( (T8) \))

Initializes a new instance of the `VariableCollection` class.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public VariableCollection(
    Func<T, T2, T3, T4, T5, T6, T7, T8, StringBuilder, double lowerBound, double upperBound, VariableType type, IEnumerable<T> set1, IEnumerable<T2> set2, IEnumerable<T3> set3, IEnumerable<T4> set4, IEnumerable<T5> set5, IEnumerable<T6> set6, IEnumerable<T7> set7, IEnumerable<T8> set8)
)
```

Visual Basic

```vbnet
Public Sub New ( __
    uniqueNamegenerator As Func(Of T, T2, T3, T4, lowerBound As Double, __
    upperBound As Double, __
    type As VariableType, __
    set1 As IEnumerable(Of T), __
    set2 As IEnumerable(Of T2), __
    set3 As IEnumerable(Of T3), __
    set4 As IEnumerable(Of T4), __
    set5 As IEnumerable(Of T5), __
    set6 As IEnumerable(Of T6), __
    set7 As IEnumerable(Of T7), __
    set8 As IEnumerable(Of T8) __
)
```

Visual C++
public:
VariableCollection(
    Func<T, T2, T3, T4, T5, T6, T7, T8, StringBuilder, 
    double lowerBound, 
    double upperBound, 
    VariableType type, 
    IEnumerable<T>^ set1, 
    IEnumerable<T2>^ set2, 
    IEnumerable<T3>^ set3, 
    IEnumerable<T4>^ set4, 
    IEnumerable<T5>^ set5, 
    IEnumerable<T6>^ set6, 
    IEnumerable<T7>^ set7, 
    IEnumerable<T8>^ set8
)

Parameters

uniqueNamegenerator
Type: System.Func(T, T2, T3, T4, T5, T6, T7, T8, StringBuilder)
A unique namegenerator.

lowerBound
Type: System.Double
The lower bound.

upperBound
Type: System.Double
The upper bound.

type
Type: Optimization.VariableType
The type.

set1
Type: System.Collections.Generic.IEnumerable(T)
The first set

set2
Type: `System.Collections.Generic.IEnumerable<T2>`
The second set  

`set3`

Type: `System.Collections.Generic.IEnumerable<T3>`
The third set  

`set4`

Type: `System.Collections.Generic.IEnumerable<T4>`
The fourth set  

`set5`

Type: `System.Collections.Generic.IEnumerable<T5>`
The fifth set  

`set6`

Type: `System.Collections.Generic.IEnumerable<T6>`
The sixth set  

`set7`

Type: `System.Collections.Generic.IEnumerable<T7>`
The seventh set  

`set8`

Type: `System.Collections.Generic.IEnumerable<T8>`
The eighth set
See Also

VariableCollection(T, T2, T3, T4, T5, T6, T7, T8) Class
VariableCollection(T, T2, T3, T4, T5, T6, T7, T8) Overload
Optimization Namespace
VariableCollection\((T, T2, T3, T4, T5, T6, T7, T8)\) Constructor (String, Double, Double, VariableType, IEnumerable\((T)\), IEnumerable\((T2)\), IEnumerable\((T3)\), IEnumerable\((T4)\), IEnumerable\((T5)\), IEnumerable\((T6)\), IEnumerable\((T7)\), IEnumerable\((T8)\))

Initializes a new instance of the `VariableCollection` class.

**Namespace:** Optimization

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public VariableCollection(
    string name,
    double lowerBound,
    double upperBound,
    VariableType type,
    IEnumerable<T> set1,
    IEnumerable<T2> set2,
    IEnumerable<T3> set3,
    IEnumerable<T4> set4,
    IEnumerable<T5> set5,
    IEnumerable<T6> set6,
    IEnumerable<T7> set7,
    IEnumerable<T8> set8
)
```

**Visual Basic**

```vbnet
Public Sub New (_
    name As String, _
    lowerBound As Double, _
    upperBound As Double, _
    type As VariableType, _
    set1 As IEnumerable(Of T), _
    set2 As IEnumerable(Of T2), _
    set3 As IEnumerable(Of T3), _
    set4 As IEnumerable(Of T4), _
    set5 As IEnumerable(Of T5), _
    set6 As IEnumerable(Of T6), _
    set7 As IEnumerable(Of T7), _
    set8 As IEnumerable(Of T8) _
)
```

**Visual C++**
public:
VariableCollection(
    String^ name,
    double lowerBound,
    double upperBound,
    VariableType type,
    IEnumerable<T>^ set1,
    IEnumerable<T2>^ set2,
    IEnumerable<T3>^ set3,
    IEnumerable<T4>^ set4,
    IEnumerable<T5>^ set5,
    IEnumerable<T6>^ set6,
    IEnumerable<T7>^ set7,
    IEnumerable<T8>^ set8
)

Parameters

name
Type: System.String
The name.

lowerBound
Type: System.Double
The lower bound.

upperBound
Type: System.Double
The upper bound.

type
Type: Optimization.VariableType
The type.

set1
Type: System.Collections.Generic(IEnumerable(T)
The first set

set2
Type: `System.Collections.Generic.IEnumerable<T2>`
The second set

`set3`
Type: `System.Collections.Generic.IEnumerable<T3>`
The third set

`set4`
Type: `System.Collections.Generic.IEnumerable<T4>`
The fourth set

`set5`
Type: `System.Collections.Generic.IEnumerable<T5>`
The fifth set

`set6`
Type: `System.Collections.Generic.IEnumerable<T6>`
The sixth set

`set7`
Type: `System.Collections.Generic.IEnumerable<T7>`
The seventh set

`set8`
Type: `System.Collections.Generic.IEnumerable<T8>`
The eighth set
See Also

VariableCollection(T, T2, T3, T4, T5, T6, T7, T8) Class

VariableCollection(T, T2, T3, T4, T5, T6, T7, T8) Overload

Optimization Namespace
The `VariableCollection(T, T2, T3, T4, T5, T6, T7, T8)` type exposes the following members.
## Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>_internalVariableCollection</code></td>
<td>(Inherited from <a href="#">GenericVariableCollectionBase</a>.)</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4, T5, T6, T7, T8) Class
Optimization Namespace
The `VariableCollection(T, T2, T3, T4, T5, T6, T7, T8)` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup</td>
</tr>
<tr>
<td></td>
<td>operations before the <code>Object</code> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <code>Type</code> of the current instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <code>Object</code>.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.</td>
</tr>
<tr>
<td><strong>SetVariableValues</strong></td>
<td>Sets the value property for each variable in this VariableCollection to</td>
</tr>
<tr>
<td></td>
<td>the corresponding value in the parameter 'values'. The matching happens</td>
</tr>
<tr>
<td></td>
<td>by string comparison of the name of the variable.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>GenericVariableCollectionBase</code>).</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4, T5, T6, T7, T8) Class
Optimization Namespace
The `VariableCollection(T, T2, T3, T4, T5, T6, T7, T8)` type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Index Validation</strong></td>
<td>Changes the behavior of a VariableCollection. If set to true the VariableCollection will validate if an index belongs to the sets the VariableCollection is based on (Inherited from <a href="#">GenericVariableCollectionBase</a>).</td>
</tr>
<tr>
<td><strong>Item</strong></td>
<td>Gets the <a href="#">Variable</a> with the specified index.</td>
</tr>
</tbody>
</table>
See Also

VariableCollection(T, T2, T3, T4, T5, T6, T7, T8) Class
Optimization Namespace
VariableCollection\(T, T2, T3, T4, T5, T6, T7, T8\).Item Property

Gets the \texttt{Variable} with the specified index.

\textbf{Namespace:} Optimization

\textbf{Assembly:} Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>public <code>Variable</code> this[</td>
<td>Public ReadOnly Default Property Item (</td>
<td>public: property `Variable^ default[T</td>
</tr>
<tr>
<td>T element1,</td>
<td>_ element1 As T, _</td>
<td>element1, T2 element2, T3 element3,</td>
</tr>
<tr>
<td>T2 element2,</td>
<td>element2 As T2, _</td>
<td>T4 element4, T5 element5, T6 element6,</td>
</tr>
<tr>
<td>T3 element3,</td>
<td>element3 As T3, _</td>
<td>T7 element7, T8 element8</td>
</tr>
<tr>
<td>T4 element4,</td>
<td>element4 As T4, _</td>
<td>] { get; }</td>
</tr>
<tr>
<td>T5 element5,</td>
<td>element5 As T5, _</td>
<td></td>
</tr>
<tr>
<td>T6 element6,</td>
<td>element6 As T6, _</td>
<td>Get</td>
</tr>
<tr>
<td>T7 element7,</td>
<td>element7 As T7, _</td>
<td></td>
</tr>
<tr>
<td>T8 element8</td>
<td>element8 As T8 _</td>
<td></td>
</tr>
<tr>
<td>] { get; }</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Parameters**

*element1*
Type: T

*element2*
Type: T2

*element3*
Type: T3

*element4*
Type: T4

*element5*
Type: T5

*element6*
Type: T6

*element7*
Type: T7

*element8*
Type: T8
See Also

VariableCollection(T, T2, T3, T4, T5, T6, T7, T8) Class
Optimization Namespace
VariableType Enumeration

Type of a IVariable.

Namespace: Optimization
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public enum VariableType</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Enumeration VariableType</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public enum class VariableType</code></td>
</tr>
</tbody>
</table>
## Members

<table>
<thead>
<tr>
<th>Member name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>0</td>
<td>Whole number.</td>
</tr>
<tr>
<td>Continuous</td>
<td>1</td>
<td>Continuous number.</td>
</tr>
</tbody>
</table>
See Also

Optimization Namespace
The classes needed to support configuring solvers via your app.config or web.config
## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ModelElement</td>
<td>Defines settings for all models you will create in within this application.</td>
</tr>
<tr>
<td>OptimizationConfigSection</td>
<td>The OptimizationConfigSection Configuration Section.</td>
</tr>
<tr>
<td>SolverElement</td>
<td>Specifies settings for a particular solver.</td>
</tr>
<tr>
<td>SolversCollection</td>
<td>The solvers collection allows you to specify settings for solvers you want to use within this application.</td>
</tr>
</tbody>
</table>
Optimization Framework

ModelElement Class

Defines settings for all models you will create in within this application.

Namespace: Optimization.Configuration

Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th></th>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public class ModelElement : ConfigurationElement</code></td>
<td><strong>Public Class ModelElement _</strong></td>
<td><code>public ref class ModelElement : public ConfigurationElement</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><strong>Inherits ConfigurationElement</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
System.Configuration.ConfigurationElement
Optimization.Configuration.ModelElement
See Also

ModelElement Members
Optimization.Configuration Namespace
The `ModelElement` type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ModelElement</code></td>
<td>Initializes a new instance of the <code>ModelElement</code> class</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeserializeElement</td>
<td>Reads XML from the configuration file. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Equals</td>
<td>Compares the current ConfigurationElement instance to the specified object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup</td>
</tr>
<tr>
<td></td>
<td>operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Gets a unique value representing the current ConfigurationElement instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>GetTransformedAssemblyString</td>
<td>Returns the transformed version of the specified assembly name.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>GetTransformedTypeString</td>
<td>Returns the transformed version of the specified type name.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>Init</td>
<td>Sets the ConfigurationElement object to its initial state.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>InitializeDefault</td>
<td>Used to initialize a default set of values for the <code>ConfigurationElement</code> object. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>IsModified</td>
<td>Indicates whether this configuration element has been modified since it was last saved or loaded, when implemented in a derived class. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>IsReadOnly</td>
<td>Gets a value indicating whether the element is read-only. ( Overrides <code>ConfigurationElement.IsReadOnly</code>.)</td>
</tr>
<tr>
<td>ListErrors</td>
<td>Adds the invalid-property errors in this <code>ConfigurationElement</code> object, and in all subelements, to the passed list. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>OnDeserializeUnrecognizedAttribute</td>
<td>Gets a value indicating whether an unknown attribute is encountered during deserialization. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>OnDeserializeUnrecognizedElement</td>
<td>Gets a value indicating whether an unknown element is encountered during deserialization. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>OnRequiredPropertyNotFound</td>
<td>Throws an exception when a required property is not found. (Inherited from</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>PostDeserialize</strong></td>
<td>Called after deserialization. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>PreSerialize</strong></td>
<td>Called before serialization. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>Reset</strong></td>
<td>Resets the internal state of the ConfigurationElement object, including the locks and the properties collections. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>ResetModified</strong></td>
<td>Resets the value of the IsModified() method to false when implemented in a derived class. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>SerializeElement</strong></td>
<td>Writes the contents of this configuration element to the configuration file when implemented in a derived class. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>SerializeToXmlElement</strong></td>
<td>Writes the outer tags of this configuration element to the configuration file when implemented in a derived class. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>SetPropertyValue</strong></td>
<td>Sets a property to the specified value. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>SetReadOnly</strong></td>
<td>Sets the IsReadOnly() property for the ConfigurationElement object and all subelements. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>Unmerge</strong></td>
<td>Modifies the <strong>ConfigurationElement</strong> object to remove all values that should not be saved. (Inherited from <strong>ConfigurationElement</strong>.)</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Behavior</strong></td>
<td>Gets or sets 'auto' will turn on automatic variable handling, i.e. when you add constraints and objectives the variables in them will be added automatically (default), 'manual' will turn this feature off and you need to add variables manually to the model</td>
</tr>
<tr>
<td><strong>CurrentConfiguration</strong></td>
<td>Gets a reference to the top-level Configuration instance that represents the configuration hierarchy that the current ConfigurationElement instance belongs to. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>ElementInformation</strong></td>
<td>Gets an ElementInformation object that contains the non-customizable information and functionality of the ConfigurationElement object. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>ElementProperty</strong></td>
<td>Gets the ConfigurationElementProperty object that represents the ConfigurationElement object itself. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>EvaluationContext</strong></td>
<td>Gets the ContextInformation object for the ConfigurationElement object. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>Item(ConfigurationProperty)</strong></td>
<td>Gets or sets a property or attribute of this configuration element.</td>
</tr>
</tbody>
</table>

**Table:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Behavior</strong></td>
<td>Gets or sets 'auto' will turn on automatic variable handling, i.e. when you add constraints and objectives the variables in them will be added automatically (default), 'manual' will turn this feature off and you need to add variables manually to the model</td>
</tr>
<tr>
<td><strong>CurrentConfiguration</strong></td>
<td>Gets a reference to the top-level Configuration instance that represents the configuration hierarchy that the current ConfigurationElement instance belongs to. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>ElementInformation</strong></td>
<td>Gets an ElementInformation object that contains the non-customizable information and functionality of the ConfigurationElement object. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>ElementProperty</strong></td>
<td>Gets the ConfigurationElementProperty object that represents the ConfigurationElement object itself. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>EvaluationContext</strong></td>
<td>Gets the ContextInformation object for the ConfigurationElement object. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>Item(ConfigurationProperty)</strong></td>
<td>Gets or sets a property or attribute of this configuration element.</td>
</tr>
<tr>
<td>Method/Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Item(String)</td>
<td>Gets or sets a property, attribute, or child element of this configuration element. (Inherited from <a href="#">ConfigurationElement</a>).</td>
</tr>
<tr>
<td>LockAllAttributesExcept</td>
<td>Gets the collection of locked attributes. (Inherited from <a href="#">ConfigurationElement</a>).</td>
</tr>
<tr>
<td>LockAllElementsExcept</td>
<td>Gets the collection of locked elements. (Inherited from <a href="#">ConfigurationElement</a>).</td>
</tr>
<tr>
<td>LockAttributes</td>
<td>Gets the collection of locked attributes (Inherited from <a href="#">ConfigurationElement</a>).</td>
</tr>
<tr>
<td>LockElements</td>
<td>Gets the collection of locked elements. (Inherited from <a href="#">ConfigurationElement</a>).</td>
</tr>
<tr>
<td>LockItem</td>
<td>Gets or sets a value indicating whether the element is locked. (Inherited from <a href="#">ConfigurationElement</a>).</td>
</tr>
<tr>
<td>Properties</td>
<td>Gets the collection of properties. (Inherited from <a href="#">ConfigurationElement</a>).</td>
</tr>
</tbody>
</table>
See Also

ModelElement Class
Optimization.Configuration Namespace
Initializes a new instance of the **ModelElement** class

**Namespace:** [Optimization.Configuration](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
</tr>
<tr>
<td>public ModelElement()</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
</tr>
<tr>
<td>Public Sub New</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
</tr>
<tr>
<td>public: ModelElement()</td>
</tr>
</tbody>
</table>
See Also

ModelElement Class
Optimization.Configuration Namespace
The ModelElement type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeserializeElement</td>
<td>Reads XML from the configuration file. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Equals</td>
<td>Compares the current ConfigurationElement instance to the specified object. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Gets a unique value representing the current ConfigurationElement instance. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>GetTransformedAssemblyString</td>
<td>Returns the transformed version of the specified assembly name. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>GetTransformedTypeString</td>
<td>Returns the transformed version of the specified type name. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td>Init</td>
<td>Sets the ConfigurationElement object to its initial state. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>InitializeDefault</td>
<td>Used to initialize a default set of values for the <a href="#">ConfigurationElement</a> object. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
<tr>
<td>IsModified</td>
<td>Indicates whether this configuration element has been modified since it was last saved or loaded, when implemented in a derived class. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
<tr>
<td>IsReadOnly</td>
<td>Gets a value indicating whether the element is read-only. (Overrides <a href="#">ConfigurationElement</a>.IsReadOnly())</td>
</tr>
<tr>
<td>ListErrors</td>
<td>Adds the invalid-property errors in this <a href="#">ConfigurationElement</a> object, and in all subelements, to the passed list. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>OnDeserializedUnrecognizedAttribute</td>
<td>Gets a value indicating whether an unknown attribute is encountered during deserialization. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
<tr>
<td>OnDeserializedUnrecognizedElement</td>
<td>Gets a value indicating whether an unknown element is encountered during deserialization. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
<tr>
<td>OnRequiredPropertyNotFound</td>
<td>Throws an exception when a required property is not found. (Inherited from</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PostDeserialize</td>
<td>Called after deserialization.</td>
</tr>
<tr>
<td>PreSerialize</td>
<td>Called before serialization.</td>
</tr>
<tr>
<td>Reset</td>
<td>Resets the internal state of the ConfigurationElement object, including the locks and the properties collections.</td>
</tr>
<tr>
<td>ResetModified</td>
<td>Resets the value of the IsModified() method to false when implemented in a derived class.</td>
</tr>
<tr>
<td>SerializeElement</td>
<td>Writes the contents of this configuration element to the configuration file when implemented in a derived class.</td>
</tr>
<tr>
<td>SerializeToXmlElement</td>
<td>Writes the outer tags of this configuration element to the configuration file when implemented in a derived class.</td>
</tr>
<tr>
<td>SetPropertyValue</td>
<td>Sets a property to the specified value.</td>
</tr>
<tr>
<td>SetReadOnly</td>
<td>Sets the IsReadOnly() property for the ConfigurationElement object and all subelements.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <em>String</em> that represents the current <em>Object</em>. (Inherited from <em>Object.</em>)</td>
</tr>
<tr>
<td><strong>Unmerge</strong></td>
<td>Modifies the <em>ConfigurationElement</em> object to remove all values that should not be saved. (Inherited from <em>ConfigurationElement.</em>)</td>
</tr>
</tbody>
</table>
See Also

ModelElement Class
Optimization.Configuration Namespace
Gets a value indicating whether the element is read-only.

**Namespace:** Optimization.Configuration  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

**C#**

```csharp
public override bool IsReadOnly()
```

**Visual Basic**

```vbnet
Public Overrides Function IsReadOnly As Boolean
```

**Visual C++**

```cpp
public:
virtual bool IsReadOnly() override
```

### Return Value

See Also

ModelElement Class
Optimization.Configuration Namespace
The ModelElement type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Behavior] Behavior</td>
<td>Gets or sets 'auto' will turn on automatic variable handling, i.e. when you add constraints and objectives the variables in them will be added automatically (default), 'manual' will turn this feature off and you need to add variables manually to the model.</td>
</tr>
<tr>
<td>![CurrentConfiguration] CurrentConfiguration</td>
<td>Gets a reference to the top-level Configuration instance that represents the configuration hierarchy that the current ConfigurationElement instance belongs to. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>![ElementInformation] ElementInformation</td>
<td>Gets an ElementInformation object that contains the non-customizable information and functionality of the ConfigurationElement object. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>![ElementProperty] ElementProperty</td>
<td>Gets the ConfigurationElementProperty object that represents the ConfigurationElement object itself. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>![EvaluationContext] EvaluationContext</td>
<td>Gets the ContextInformation object for the ConfigurationElement object. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>![Item(ConfigurationProperty)] Item(ConfigurationProperty)</td>
<td>Gets or sets a property or attribute of this configuration element.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Item(String)</td>
<td>Gets or sets a property, attribute, or child element of this configuration element. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>LockAllAttributesExcept</td>
<td>Gets the collection of locked attributes. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>LockAllElementsExcept</td>
<td>Gets the collection of locked elements. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>LockAttributes</td>
<td>Gets the collection of locked attributes (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>LockElements</td>
<td>Gets the collection of locked elements. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>LockItem</td>
<td>Gets or sets a value indicating whether the element is locked. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Properties</td>
<td>Gets the collection of properties. (Inherited from ConfigurationElement.)</td>
</tr>
</tbody>
</table>
See Also

- ModelElement Class
- Optimization.Configuration Namespace
Gets or sets 'auto' will turn on automatic variable handling, i.e. when you add constraints and objectives the variables in them will be added automatically (default), 'manual' will turn this feature off and you need to add variables manually to the model.

**Namespace:** Optimization.Configuration  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

#### C#

```csharp
public string Behavior { get; set; }
```

#### Visual Basic

```vbnet
Public Property Behavior As String
    Get
        Get
    Set
```

#### Visual C++

```cpp
public:
    property String^ Behavior {
        String^ get ();
        void set (String^ value);
    }
```
See Also

ModelElement Class
Optimization.Configuration Namespace
ModelElement.Item Property
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item(ConfigurationProperty)</td>
<td>Gets or sets a property or attribute of this configuration element. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Item(String)</td>
<td>Gets or sets a property, attribute, or child element of this configuration element. (Inherited from ConfigurationElement.)</td>
</tr>
</tbody>
</table>
See Also

ModelElement Class
ModelElement Members
Optimization.Configuration Namespace
The OptimizationConfigSection Configuration Section.

**Namespace:** Optimization.Configuration

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public class OptimizationConfigSection : ConfigurationSection</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Class OptimizationConfigSection _ Inherits ConfigurationSection</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public ref class OptimizationConfigSection : public ConfigurationSection</code></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
System.Configuration.ConfigurationElement
System.Configuration.ConfigurationSection
Optimization.Configuration.OptimizationConfigSection
See Also

- OptimizationConfigSection Members
- Optimization.Configuration Namespace
Optimization Framework

**OptimizationConfigSection Members**

[OptimizationConfigSection Class Constructors Methods Properties See Also Send Feedback]

The **OptimizationConfigSection** type exposes the following members.
# Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OptimizationConfigSection</td>
<td>Initializes a new instance of the OptimizationConfigSection class</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeserializeElement</td>
<td>Reads XML from the configuration file. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>DeserializeSection</td>
<td>Reads XML from the configuration file. (Inherited from ConfigurationSection.)</td>
</tr>
<tr>
<td>Equals</td>
<td>Compares the current ConfigurationElement instance to the specified object. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Gets a unique value representing the current ConfigurationElement instance. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>GetRuntimeObject</td>
<td>Returns a custom object when overridden in a derived class. (Inherited from ConfigurationSection.)</td>
</tr>
<tr>
<td>GetTransformedAssemblyString</td>
<td>Returns the transformed version of the specified assembly name. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>GetTransformedTypeString</td>
<td>Returns the transformed version of the specified type name.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>GetTypeInfo</code></td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><code>Init</code></td>
<td>Sets the <code>ConfigurationElement</code> object to its initial state. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><code>InitializeDefault</code></td>
<td>Used to initialize a default set of values for the <code>ConfigurationElement</code> object. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><code>IsModified</code></td>
<td>Indicates whether this configuration element has been modified since was last saved or loaded when implemented in a derived class. (Inherited from <code>ConfigurationSection</code>.)</td>
</tr>
<tr>
<td><code>IsReadOnly</code></td>
<td>Gets a value indicating whether the element is read-only. (Overrides <code>ConfigurationElement.IsReadOnly</code>.)</td>
</tr>
<tr>
<td><code>ListErrors</code></td>
<td>Adds the invalid-property errors in this <code>ConfigurationElement</code> object and in all subelements, to the passed list. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><code>MemberwiseClone</code></td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><code>OnDeserializeUnrecognizedAttribute</code></td>
<td>Gets a value indicating whether an unknown attribute is encountered during deserialization. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>OnDeserializeUnrecognizedElement</td>
<td>Gets a value indicating whether an unknown element is encountered during deserialization. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>OnRequiredPropertyNotFound</td>
<td>Throws an exception when a required property is not found. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>PostDeserialize</td>
<td>Called after deserialization. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>PreSerialize</td>
<td>Called before serialization. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Reset</td>
<td>Resets the internal state of the ConfigurationElement object, including the locks and the property collections. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>ResetModified</td>
<td>Resets the value of the IsModified method to false when implemented in a derived class. (Inherited from ConfigurationSection.)</td>
</tr>
<tr>
<td>SerializeElement</td>
<td>Writes the contents of this configuration element to the configuration file when implemented in a derived class. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>SerializeSection</td>
<td>Creates an XML string containing an unmerged view of the ConfigurationSection object as a single section to write to a file. (Inherited from ConfigurationSection.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SerializeToXmlElement</td>
<td>Writes the outer tags of this configuration element to the configuration file when implemented in a derived class. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>SetPropertyValue</td>
<td>Sets a property to the specified value. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>SetReadOnly</td>
<td>Sets the IsReadOnly() property for the ConfigurationElement object and all subelements. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>ShouldSerializeElementInTargetVersion</td>
<td>Indicates whether the specified element should be serialized when the configuration object hierarchy is serialized for the specified target version of the .NET Framework. (Inherited from ConfigurationSection.)</td>
</tr>
<tr>
<td>ShouldSerializePropertyInTargetVersion</td>
<td>Indicates whether the specified property should be serialized when the configuration object hierarchy is serialized for the specified target version of the .NET Framework. (Inherited from ConfigurationSection.)</td>
</tr>
<tr>
<td>ShouldSerializeSectionInTargetVersion</td>
<td>Indicates whether the current ConfigurationSection instance should be serialized when the configuration object hierarchy is serialized for the specified target version of the .NET Framework. (Inherited from ConfigurationSection.)</td>
</tr>
<tr>
<td><strong>Unmerge</strong></td>
<td>Modifies the <a href="https://learn.microsoft.com/en-us/dotnet/api/system.configuration.configurationelement">ConfigurationElement</a> object to remove all values that should not be saved. (Inherited from <a href="https://learn.microsoft.com/en-us/dotnet/api/system.configuration.configurationelement">ConfigurationElement</a>.)</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CurrentConfiguration</strong></td>
<td>Gets a reference to the top-level <a href="#">Configuration</a> instance that represents the configuration hierarchy that the current <a href="#">ConfigurationElement</a> instance belongs to. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
<tr>
<td><strong>ElementInformation</strong></td>
<td>Gets an <a href="#">ElementInformation</a> object that contains the non-customizable information and functionality of the <a href="#">ConfigurationElement</a> object. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
<tr>
<td><strong>ElementProperty</strong></td>
<td>Gets the <a href="#">ConfigurationElementProperty</a> object that represents the <a href="#">ConfigurationElement</a> object itself. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
<tr>
<td><strong>EvaluationContext</strong></td>
<td>Gets the <a href="#">ContextInformation</a> object for the <a href="#">ConfigurationElement</a> object. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
<tr>
<td><strong>Instance</strong></td>
<td>Gets the OptimizationConfigSection instance.</td>
</tr>
<tr>
<td><strong>Item(ConfigurationProperty)</strong></td>
<td>Gets or sets a property or attribute of this configuration element. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
<tr>
<td><strong>Item(String)</strong></td>
<td>Gets or sets a property, attribute, or child element of this configuration element.</td>
</tr>
<tr>
<td>Method/Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| [LockAllAttributesExcept](#) | Gets the collection of locked attributes.  
(Inherited from [ConfigurationElement](#).) |
| [LockAllElementsExcept](#) | Gets the collection of locked elements.  
(Inherited from [ConfigurationElement](#).) |
| [LockAttributes](#) | Gets the collection of locked attributes  
(Inherited from [ConfigurationElement](#).) |
| [LockElements](#) | Gets the collection of locked elements.  
(Inherited from [ConfigurationElement](#).) |
| [LockItem](#) | Gets or sets a value indicating whether the element is locked.  
(Inherited from [ConfigurationElement](#).) |
| [ModelElement](#) | Gets or sets defines settings for all models you will create within this application. |
| [Properties](#) | Gets the collection of properties.  
(Inherited from [ConfigurationElement](#).) |
| [SectionInformation](#) | Gets a [SectionInformation](#) object that contains the non-customizable information and functionality of the [ConfigurationSection](#) object.  
(Inherited from [ConfigurationSection](#).) |
| [SolverItems](#) | Gets or sets the solvers collection allows you to specify settings for solvers you want to use within this application. |
| **Xmlns** | Gets the XML namespace of this Configuration Section. |
See Also

OptimizationConfigSection Class
Optimization.Configuration Namespace
Optimization Framework

OptimizationConfigSection Constructor

[OptimizationConfigSection Class See Also Send Feedback]

Initializes a new instance of the OptimizationConfigSection class

**Namespace:** Optimization.Configuration

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public OptimizationConfigSection()</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Sub New</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public: OptimizationConfigSection()</code></td>
</tr>
</tbody>
</table>
See Also

OptimizationConfigSection Class
Optimization.Configuration Namespace
The `OptimizationConfigSection` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeserializeElement</td>
<td>Reads XML from the configuration file. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>DeserializeSection</td>
<td>Reads XML from the configuration file. (Inherited from <code>ConfigurationSection</code>.)</td>
</tr>
<tr>
<td>Equals</td>
<td>Compares the current <code>ConfigurationElement</code> instance to the specified object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup</td>
</tr>
<tr>
<td></td>
<td>operations before the <code>Object</code> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Gets a unique value representing the current <code>ConfigurationElement</code> instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>GetRuntimeObject</td>
<td>Returns a custom object when overridden in a derived class.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>ConfigurationSection</code>.)</td>
</tr>
<tr>
<td>GetTransformedAssemblyString</td>
<td>Returns the transformed version of the specified assembly name.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>GetTransformedTypeString</td>
<td>Returns the transformed version of the specified type name.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Get Type</strong></td>
<td>Gets the Type of the current instance. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>Init</strong></td>
<td>Sets the ConfigurationElement object to its initial state. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>InitializeDefault</strong></td>
<td>Used to initialize a default set of values for the ConfigurationElement object. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>Is Modified</strong></td>
<td>Indicates whether this configuration element has been modified since it was last saved or loaded when implemented in a derived class. (Inherited from ConfigurationSection.)</td>
</tr>
<tr>
<td><strong>Is ReadOnly</strong></td>
<td>Gets a value indicating whether the element is read-only. (Overrides ConfigurationElement.IsReadOnly.)</td>
</tr>
<tr>
<td><strong>List Errors</strong></td>
<td>Adds the invalid-property errors in this ConfigurationElement object, and in all subelements, to the passed list. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>Memberwise Clone</strong></td>
<td>Creates a shallow copy of the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>On Deserialized Unrecognized Attribute</strong></td>
<td>Gets a value indicating whether an unknown attribute is encountered during deserialization. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>OnDeserializeUnrecognizedElement</td>
<td>Gets a value indicating whether an unknown element is encountered during deserialization. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>OnRequiredPropertyNotFound</td>
<td>Throws an exception when a required property is not found. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>PostDeserialize</td>
<td>Called after deserialization. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>PreSerialize</td>
<td>Called before serialization. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Reset</td>
<td>Resets the internal state of the ConfigurationElement object, including the locks and the property collections. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>ResetModified</td>
<td>Resets the value of the IsModified method to false when implemented in a derived class. (Inherited from ConfigurationSection.)</td>
</tr>
<tr>
<td>SerializeElement</td>
<td>Writes the contents of this configuration element to the configuration file when implemented in a derived class. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>SerializeSection</td>
<td>Creates an XML string containing an unmerged view of the ConfigurationSection object as a single section to write to a file. (Inherited from ConfigurationSection.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SerializeToXmlElement</td>
<td>Writes the outer tags of this configuration element to the configuration file when implemented in a derived class. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>SetPropertyValue</td>
<td>Sets a property to the specified value.                                                       (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>SetReadOnly</td>
<td>Sets the IsReadOnly() property for the ConfigurationElement object and all subelements.       (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>ShouldSerializeElementInTargetVersion</td>
<td>Indicates whether the specified element should be serialized when the configuration object hierarchy is serialized for the specified target version of the .NET Framework. (Inherited from ConfigurationSection.)</td>
</tr>
<tr>
<td>ShouldSerializePropertyInTargetVersion</td>
<td>Indicates whether the specified property should be serialized when the configuration object hierarchy is serialized for the specified target version of the .NET Framework. (Inherited from ConfigurationSection.)</td>
</tr>
<tr>
<td>ShouldSerializeSectionInTargetVersion</td>
<td>Indicates whether the current ConfigurationSection instance should be serialized when the configuration object hierarchy is serialized for the specified target version of the .NET Framework. (Inherited from ConfigurationSection.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>Unmerge</strong></td>
<td>Modifies the <strong>ConfigurationElement</strong> object to remove all values that should not be saved. (Inherited from <strong>ConfigurationElement</strong>.)</td>
</tr>
</tbody>
</table>
See Also

- OptimizationConfigSection Class
- Optimization.Configuration Namespace
Optimization Framework

OptimizationConfigSection.IsReadOnly Method

See Also
Send Feedback

Gets a value indicating whether the element is read-only.

**Namespace:** Optimization.Configuration

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public override bool IsReadOnly()</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Overrides Function IsReadOnly As Boolean</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public: virtual bool IsReadOnly() override</code></td>
</tr>
</tbody>
</table>

**Return Value**

See Also

OptimizationConfigSection Class
Optimization.Configuration Namespace
The **OptimizationConfigSection** type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| **CurrentConfiguration**    | Gets a reference to the top-level [Configuration](#) instance that represents the configuration hierarchy that the current [ConfigurationElement](#) instance belongs to.  
(Inherited from [ConfigurationElement](#).) |
| **ElementInformation**      | Gets an [ElementInformation](#) object that contains the non-customizable information and functionality of the [ConfigurationElement](#) object.  
(Inherited from [ConfigurationElement](#).) |
| **ElementProperty**         | Gets the [ConfigurationElementProperty](#) object that represents the [ConfigurationElement](#) object itself.  
(Inherited from [ConfigurationElement](#).) |
| **EvaluationContext**       | Gets the [ContextInformation](#) object for the [ConfigurationElement](#) object.  
(Inherited from [ConfigurationElement](#).) |
| **Instance**                | Gets the OptimizationConfigSection instance.                                                                                       |
| **Item(ConfigurationProperty)** | Gets or sets a property or attribute of this configuration element.  
(Inherited from [ConfigurationElement](#).) |
<p>| <strong>Item(String)</strong>             | Gets or sets a property, attribute, or child element of this configuration element.                                                       |</p>
<table>
<thead>
<tr>
<th>Method/Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LockAllAttributesExcept</strong></td>
<td>Gets the collection of locked attributes. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>LockAllElementsExcept</strong></td>
<td>Gets the collection of locked elements. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>LockAttributes</strong></td>
<td>Gets the collection of locked attributes. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>LockElements</strong></td>
<td>Gets the collection of locked elements. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>LockItem</strong></td>
<td>Gets or sets a value indicating whether the element is locked. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>ModelElement</strong></td>
<td>Gets or sets defines settings for all models you will create within this application.</td>
</tr>
<tr>
<td><strong>Properties</strong></td>
<td>Gets the collection of properties. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>SectionInformation</strong></td>
<td>Gets a <code>SectionInformation</code> object that contains the non-customizable information and functionality of the <code>ConfigurationSection</code> object. (Inherited from <code>ConfigurationSection</code>.)</td>
</tr>
<tr>
<td><strong>SolverItems</strong></td>
<td>Gets or sets the solvers collection allows you to specify settings for solvers you want to use within this application.</td>
</tr>
<tr>
<td><strong>XmlNs</strong></td>
<td>Gets the XML namespace of this Configuration Section.</td>
</tr>
</tbody>
</table>
See Also

OptimizationConfigSection Class
Optimization.Configuration Namespace
Gets the OptimizationConfigSection instance.

**Namespace:** Optimization.Configuration  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
<tr>
<td>public static <a href="#">OptimizationConfigSection</a> Instance { get</td>
<td></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td></td>
</tr>
<tr>
<td>Public Shared ReadOnly Property Instance As <a href="#">OptimizationConfigSection</a> Get</td>
<td></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
</tr>
<tr>
<td>public:</td>
<td></td>
</tr>
<tr>
<td>static property <a href="#">OptimizationConfigSection</a>^ Instance .</td>
<td></td>
</tr>
<tr>
<td><a href="#">OptimizationConfigSection</a>^ get ();</td>
<td></td>
</tr>
</tbody>
</table>

---

[OptimizationConfigSection](#): A placeholder for the actual class name in the context of the code snippet.
See Also

OptimizationConfigSection Class
Optimization.Configuration Namespace
Optimization Framework

OptimizationConfigSection.Item Property

OptimizationConfigSection Class See Also Send Feedback
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item(ConfigurationProperty)</td>
<td>Gets or sets a property or attribute of this configuration element. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Item(String)</td>
<td>Gets or sets a property, attribute, or child element of this configuration element. (Inherited from ConfigurationElement.)</td>
</tr>
</tbody>
</table>
See Also

OptimizationConfigSection Class
OptimizationConfigSection Members
Optimization.Configuration Namespace
Optimization Config Section Model Element Property

Gets or sets defines settings for all models you will create within this application.

Namespace: Optimization.Configuration
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public ModelElement ModelElement { get; set; }</code></td>
</tr>
</tbody>
</table>
| **Visual Basic** | `Public Property ModelElement As ModelElement`  
`Get`  
`Set` |
| **Visual C++** | `public:  
property ModelElement^ ModelElement {  
    ModelElement^ get ();  
    void set (ModelElement^ value);  
}` |
See Also

OptimizationConfigSection Class
Optimization.Configuration Namespace
Gets or sets the solvers collection allows you to specify settings for solvers you want to use within this application.

**Namespace:** [Optimization.Configuration](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public SolversCollection SolverItems { get; set; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Property SolverItems As SolversCollection</td>
</tr>
<tr>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>Set</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: property SolversCollection^ SolverItems {</td>
</tr>
<tr>
<td></td>
<td>SolversCollection^ get ();</td>
</tr>
<tr>
<td></td>
<td>void set (SolversCollection^ value);</td>
</tr>
</tbody>
</table>
See Also

OptimizationConfigSection Class
Optimization.Configuration Namespace
Gets the XML namespace of this Configuration Section.

**Namespace**: [Optimization.Configuration](#)

**Assembly**: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
</tr>
<tr>
<td>public string Xmlns { get; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
</tr>
<tr>
<td>Public ReadOnly Property Xmlns As String Get</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
</tr>
<tr>
<td>public: property String^ Xmlns { String^ get (); }</td>
</tr>
</tbody>
</table>
Remarks

This property makes sure that if the configuration file contains the XML namespace, the parser doesn't throw an exception because it encounters the unknown "xmlns" attribute.
See Also

- OptimizationConfigSection Class
- Optimization.Configuration Namespace
SolverElement Class

Specifies settings for a particular solver.

**Namespace:** Optimization.Configuration  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public class SolverElement : ConfigurationElement</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Class SolverElement _ Inherits ConfigurationElement</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public ref class SolverElement : public ConfigurationElement</td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

- `System.Object`
- `System.Configuration.ConfigurationElement`
- `Optimization.Configuration.SolverElement`
See Also

SolverElement Members
Optimization.Configuration Namespace
The **SolverElement** type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SolverElement</td>
<td>Initializes a new instance of the SolverElement class</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeserializeElement</td>
<td>Reads XML from the configuration file. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Equals</td>
<td>Compares the current ConfigurationElement instance to the specified object. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Gets a unique value representing the current ConfigurationElement instance. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>GetTransformedAssemblyString</td>
<td>Returns the transformed version of the specified assembly name. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>GetTransformedTypeString</td>
<td>Returns the transformed version of the specified type name. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td>Init</td>
<td>Sets the ConfigurationElement object to its initial state. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>InitializeDefault</strong></td>
<td>Used to initialize a default set of values for the <code>ConfigurationElement</code> object. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>IsModified</strong></td>
<td>Indicates whether this configuration element has been modified since it was last saved or loaded, when implemented in a derived class. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>IsReadOnly</strong></td>
<td>Gets a value indicating whether the element is read-only. (Overrides <code>ConfigurationElement.IsReadOnly</code>.)</td>
</tr>
<tr>
<td><strong>ListErrors</strong></td>
<td>Adds the invalid-property errors in this <code>ConfigurationElement</code> object, and in all subelements, to the passed list. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>OnDeserialzeUnrecognizedAttribute</strong></td>
<td>Gets a value indicating whether an unknown attribute is encountered during deserialization. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>OnDeserialzeUnrecognizedElement</strong></td>
<td>Gets a value indicating whether an unknown element is encountered during deserialization. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>OnRequiredPropertyNotFound</strong></td>
<td>Throws an exception when a required property is not found. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>PostDeserialize</strong></td>
<td>Called after deserialization. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><strong>PreSerialize</strong></td>
<td>Called before serialization. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><strong>Reset</strong></td>
<td>Resets the internal state of the <a href="#">ConfigurationElement</a> object, including the locks and the properties collections. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><strong>ResetModified</strong></td>
<td>Resets the value of the <a href="#">IsModified()</a> method to false when implemented in a derived class. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><strong>SerializeElement</strong></td>
<td>Writes the contents of this configuration element to the configuration file when implemented in a derived class. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><strong>SerializeToXmlElement</strong></td>
<td>Writes the outer tags of this configuration element to the configuration file when implemented in a derived class. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><strong>SetPropertyValue</strong></td>
<td>Sets a property to the specified value. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><strong>SetReadOnly</strong></td>
<td>Sets the <a href="#">IsReadOnly()</a> property for the <a href="#">ConfigurationElement</a> object and all subelements. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>Unmerge</strong></td>
<td>Modifies the <strong>ConfigurationElement</strong> object to remove all values that should not be saved. (Inherited from <strong>ConfigurationElement</strong>.)</td>
</tr>
</tbody>
</table>
Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CurrentConfiguration</td>
<td>Gets a reference to the top-level Configuration instance that represents the configuration hierarchy that the current ConfigurationElement instance belongs to. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>ElementInformation</td>
<td>Gets an ElementInformation object that contains the non-customizable information and functionality of the ConfigurationElement object. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>ElementProperty</td>
<td>Gets the ConfigurationElementProperty object that represents the ConfigurationElement object itself. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>EvaluationContext</td>
<td>Gets the ContextInformation object for the ConfigurationElement object. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Item(ConfigurationProperty)</td>
<td>Gets or sets a property or attribute of this configuration element. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Item(String)</td>
<td>Gets or sets a property, attribute, or child element of this configuration element. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>LockAllAttributesExcept</code></td>
<td>Gets the collection of locked attributes. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><code>LockAllElementsExcept</code></td>
<td>Gets the collection of locked elements. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><code>LockAttributes</code></td>
<td>Gets the collection of locked attributes (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><code>LockElements</code></td>
<td>Gets the collection of locked elements. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><code>LockItem</code></td>
<td>Gets or sets a value indicating whether the element is locked. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><code>Name</code></td>
<td>Gets or sets the name of a specific solver (&quot;Cplex&quot; is the default)</td>
</tr>
<tr>
<td><code>Path</code></td>
<td>Gets or sets specifies the path to the native dll of this solver. You can also use relative paths, like &quot;..\cplex.dll&quot;</td>
</tr>
<tr>
<td><code>Properties</code></td>
<td>Gets the collection of properties. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
</tbody>
</table>
See Also

SolverElement Class
Optimization.Configuration Namespace
Initializes a new instance of the `SolverElement` class

**Namespace:** [Optimization.Configuration](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public SolverElement()</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Sub New</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public: SolverElement()</code></td>
</tr>
</tbody>
</table>
See Also

SolverElement Class
Optimization.Configuration Namespace
The `SolverElement` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeserializeElement</td>
<td>Reads XML from the configuration file. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Equals</td>
<td>Compares the current ConfigurationElement instance to the specified object. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Gets a unique value representing the current ConfigurationElement instance. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>GetTransformedAssemblyString</td>
<td>Returns the transformed version of the specified assembly name. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>GetTransformedTypeString</td>
<td>Returns the transformed version of the specified type name. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td>Init</td>
<td>Sets the ConfigurationElement object to its initial state. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>InitializeDefault</td>
<td>Used to initialize a default set of values for the <code>ConfigurationElement</code> object. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>IsModified</td>
<td>Indicates whether this configuration element has been modified since it was last saved or loaded, when implemented in a derived class. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>IsReadOnly</td>
<td>Gets a value indicating whether the element is read-only. (Overrides <code>ConfigurationElement.IsReadOnly</code>.)</td>
</tr>
<tr>
<td>ListErrors</td>
<td>Adds the invalid-property errors in this <code>ConfigurationElement</code> object, and in all subelements, to the passed list. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>OnDeserializeUnrecognizedAttribute</td>
<td>Gets a value indicating whether an unknown attribute is encountered during deserialization. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>OnDeserializeUnrecognizedElement</td>
<td>Gets a value indicating whether an unknown element is encountered during deserialization. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>OnRequiredPropertyNotFound</td>
<td>Throws an exception when a required property is not found. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PostDeserialize</td>
<td>Called after deserialization. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>PreSerialize</td>
<td>Called before serialization. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Reset</td>
<td>Resets the internal state of the ConfigurationElement object, including the locks and the properties collections. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>ResetModified</td>
<td>Resets the value of the IsModified() method to false when implemented in a derived class.</td>
</tr>
<tr>
<td>SerializeElement</td>
<td>Writes the contents of this configuration element to the configuration file when implemented in a derived class. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>SerializeToXmlElement</td>
<td>Writes the outer tags of this configuration element to the configuration file when implemented in a derived class. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>SetPropertyValue</td>
<td>Sets a property to the specified value. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>SetReadOnly</td>
<td>Sets the IsReadOnly() property for the ConfigurationElement object and all subelements.</td>
</tr>
</tbody>
</table>

*Inherited from ConfigurationElement.*
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>Unmerge</strong></td>
<td>Modifies the <code>ConfigurationElement</code> object to remove all values that should not be saved. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
</tbody>
</table>
See Also

SolverElement Class
Optimization.Configuration Namespace
Gets a value indicating whether the element is read-only.

**Namespace:** Optimization.Configuration  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#**   | ```
public override bool IsReadOnly()
``` |
| **Visual Basic** | ```
Public Overrides Function IsReadOnly As Boolean
``` |
| **Visual C++** | ```
public:
virtual bool IsReadOnly() override
``` |

## Return Value

See Also

SolverElement Class
Optimization.Configuration Namespace
The **SolverElement** type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CurrentConfiguration</td>
<td>Gets a reference to the top-level <code>Configuration</code> instance that represents the configuration hierarchy that the current <code>ConfigurationElement</code> instance belongs to. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>ElementInformation</td>
<td>Gets an <code>ElementInformation</code> object that contains the non-customizable information and functionality of the <code>ConfigurationElement</code> object. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>ElementProperty</td>
<td>Gets the <code>ConfigurationElementProperty</code> object that represents the <code>ConfigurationElement</code> object itself. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>EvaluationContext</td>
<td>Gets the <code>ContextInformation</code> object for the <code>ConfigurationElement</code> object. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>Item(ConfigurationProperty)</td>
<td>Gets or sets a property or attribute of this configuration element. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>Item(String)</td>
<td>Gets or sets a property, attribute, or child element of this configuration element. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>LockAllAttributesExcept</td>
<td>Gets the collection of locked attributes. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
<tr>
<td>LockAllElementsExcept</td>
<td>Gets the collection of locked elements. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
<tr>
<td>LockAttributes</td>
<td>Gets the collection of locked attributes. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
<tr>
<td>LockElements</td>
<td>Gets the collection of locked elements. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
<tr>
<td>LockItem</td>
<td>Gets or sets a value indicating whether the element is locked. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
<tr>
<td>Name</td>
<td>Gets or sets the name of a specific solver (&quot;Cplex&quot; is the default)</td>
</tr>
<tr>
<td>Path</td>
<td>Gets or sets specifies the path to the native dll of this solver. You can also use relative paths, like &quot;....\cplex.dll&quot;</td>
</tr>
<tr>
<td>Properties</td>
<td>Gets the collection of properties. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
</tbody>
</table>
See Also

SolverElement Class
Optimization.Configuration Namespace
Optimization Framework

**SolverElement.Item Property**

[SolverElement Class] [See Also] [Send Feedback]
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Item(ConfigurationProperty)</strong></td>
<td>Gets or sets a property or attribute of this configuration element. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td><strong>Item(String)</strong></td>
<td>Gets or sets a property, attribute, or child element of this configuration element. (Inherited from ConfigurationElement.)</td>
</tr>
</tbody>
</table>
See Also

SolverElement Class
SolverElement Members
Optimization.Configuration Namespace
SolverElement.Name Property

Namespace: Optimization.Configuration
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0

Gets or sets the name of a specific solver ("Cplex" is the default)
<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
</tr>
<tr>
<td>public string Name { get; set; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
</tr>
<tr>
<td>Public Property Name As String</td>
</tr>
<tr>
<td>Get</td>
</tr>
<tr>
<td>Set</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
</tr>
<tr>
<td>public: property String^ Name {</td>
</tr>
<tr>
<td>String^ get ();</td>
</tr>
<tr>
<td>void set (String^ value);</td>
</tr>
</tbody>
</table>
| }
See Also

SolverElement Class
Optimization.Configuration Namespace
Gets or sets specifies the path to the native dll of this solver. You can also use relative paths, like "..\..\cplex.dll"

**Namespace:** [Optimization.Configuration](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#** | ```
public string Path { get; set; }
``` |
| **Visual Basic** | ```
Public Property Path As String
    Get
    Set
``` |
| **Visual C++** | ```
public:
    property String^ Path {
        String^ get ();
        void set (String^ value);
    }
``` |
See Also

SolverElement Class
Optimization.Configuration Namespace
The solvers collection allows you to specify settings for solvers you want to use within this application.

**Namespace:** [Optimization.Configuration](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public class SolversCollection : ConfigurationElementCollection</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Class SolversCollection  _  Inherits ConfigurationElementCollection</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public ref class SolversCollection : public ConfigurationElementCollection</td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
System.Configuration.ConfigurationElement
System.Configuration.ConfigurationElementCollection
Optimization.Configuration.SolversCollection
See Also

SolversCollection Members
Optimization.Configuration Namespace
Optimization Framework

**SolversCollection Members**

[SolversCollection Class] [Constructors] [Methods] [Properties] [See Also] [Send Feedback]

The **SolversCollection** type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SolversCollection</td>
<td>Initializes a new instance of the SolversCollection class</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Add" /> Add</td>
<td>Adds the specified <a href="#">SolverElement</a> to the <a href="#">ConfigurationElementCollection</a>.</td>
</tr>
<tr>
<td><img src="image" alt="BaseAdd(ConfigurationElement)" /> BaseAdd(ConfigurationElement)</td>
<td>Adds a configuration element to the <a href="#">ConfigurationElementCollection</a>. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><img src="image" alt="BaseAdd(ConfigurationElement, Boolean)" /> BaseAdd(ConfigurationElement, Boolean)</td>
<td>Adds a configuration element to the <a href="#">ConfigurationElementCollection</a>. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><img src="image" alt="BaseAdd(Int32, ConfigurationElement)" /> BaseAdd(Int32, ConfigurationElement)</td>
<td>Adds a configuration element to the <a href="#">ConfigurationElementCollection</a>. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><img src="image" alt="BaseClear" /> BaseClear</td>
<td>Removes all configuration element objects from the collection. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><img src="image" alt="BaseGet(Object)" /> BaseGet(Object)</td>
<td>Returns the configuration element with the specified key. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><img src="image" alt="BaseGet(Int32)" /> BaseGet(Int32)</td>
<td>Gets the configuration element at the specified index. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><img src="image" alt="BaseGetAllKeys" /> BaseGetAllKeys</td>
<td>Returns an array of the keys for all of the configuration elements contained in the <a href="#">ConfigurationElementCollection</a>. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><img src="image" alt="BaseGetKey" /> BaseGetKey</td>
<td>Gets the key for the <a href="#">ConfigurationElement</a>. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><img src="image" alt="BaseIndexOf" /> BaseIndexOf</td>
<td>The index of the specified <a href="#">ConfigurationElement</a>. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><img src="image" alt="BaseIsRemoved" /> BaseIsRemoved</td>
<td>Gets a value indicating whether the specified key has been removed from the <a href="#">ConfigurationElementCollection</a>. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><img src="image" alt="BaseRemove" /> BaseRemove</td>
<td>Removes a <a href="#">ConfigurationElement</a> from the <a href="#">ConfigurationElementCollection</a>. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><img src="image" alt="BaseRemoveAt" /> BaseRemoveAt</td>
<td>Removes the <a href="#">ConfigurationElement</a> at the specified index. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><img src="image" alt="CopyTo" /> CopyTo</td>
<td>Copies the contents of the <a href="#">ConfigurationElement</a>. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td><img src="image" alt="CreateNewElement()" /> CreateNewElement()</td>
<td>Creates a new <a href="#">SolverElement</a>. (Overrides <a href="#">ConfigurationElementCollection</a>)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>CreateNewElement(String)</td>
<td>Creates a new ConfigurationElement (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>DeserializerElement</td>
<td>Reads XML from the configuration file. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>Equals</td>
<td>Compares the ConfigurationElement. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetElementKey</td>
<td>Gets the element key for the specified ConfigurationElement. (Overrides ConfigurationElementCollection.GetElementKey(ConfigurationElement).)</td>
</tr>
<tr>
<td>GetEnumerator</td>
<td>Gets an IEnumerator which is used to iterate through the ConfigurationElementCollection. (Inherited from ConfigurationElementCollection.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Gets a unique value representing the instance. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>GetItemAt</td>
<td>Gets the SolverElement at the specified index.</td>
</tr>
<tr>
<td>GetItemByKey</td>
<td>Gets the SolverElement with the specified key.</td>
</tr>
<tr>
<td>GetTransformedAssemblyString</td>
<td>Returns the transformed version of the specified assembly name. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>GetTransformedTypeString</td>
<td>Returns the transformed version of the specified type name. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td>Init</td>
<td>Sets the ConfigurationElement object to its initial state. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>InitializeDefault</td>
<td>Used to initialize a default set of values for the object. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>IsElementName</td>
<td>Indicates whether the specified ConfigurationElementCollection is the same as the specified ConfigurationElementCollection. (Overrides ConfigurationElementCollection.IsElementName(ConfigurationElement).)</td>
</tr>
</tbody>
</table>
| IsElementRemovable | Gets a value indicating whether the
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IsModified</strong></td>
<td>Indicates whether this <code>ConfigurationElementCollection</code> since it was last saved or loaded when overridden in a derived class. (Inherited from <code>ConfigurationElementCollection</code>)</td>
</tr>
<tr>
<td><strong>IsReadOnly</strong></td>
<td>Gets a value indicating whether the element is read-only. (Overides <code>ConfigurationElementCollection</code>)</td>
</tr>
<tr>
<td><strong>ListErrors</strong></td>
<td>Adds the invalid-property errors in this <code>ConfigurationElementCollection</code> in all subelements, to the passed list. (Inherited from <code>ConfigurationElementCollection</code>)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <code>ConfigurationElementCollection</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>OnDeserializeUnrecognizedAttribute</strong></td>
<td>Gets a value indicating whether an unknown attribute is encountered during deserialization. (Inherited from <code>ConfigurationElement</code>)</td>
</tr>
<tr>
<td><strong>OnDeserializeUnrecognizedElement</strong></td>
<td>Causes the configuration system to throw an exception. (Inherited from <code>ConfigurationElement</code>)</td>
</tr>
<tr>
<td><strong>OnRequiredPropertyNotFound</strong></td>
<td>Throws an exception when a required property is not found. (Inherited from <code>ConfigurationElement</code>)</td>
</tr>
<tr>
<td><strong>PostDeserialize</strong></td>
<td>Called after deserialization. (Inherited from <code>ConfigurationElement</code>)</td>
</tr>
<tr>
<td><strong>PreSerialize</strong></td>
<td>Called before serialization. (Inherited from <code>ConfigurationElement</code>)</td>
</tr>
<tr>
<td><strong>Remove</strong></td>
<td>Removes the specified <code>SolverElement</code> configuration element collection. (Inherited from <code>ConfigurationElementCollection</code>)</td>
</tr>
<tr>
<td><strong>Reset</strong></td>
<td>Resets the <code>ConfigurationElementCollection</code> overridden in a derived class. (Inherited from <code>ConfigurationElement</code>)</td>
</tr>
<tr>
<td><strong>ResetModified</strong></td>
<td>Resets the value of the <code>IsModified()</code> derived class. (Inherited from <code>ConfigurationElement</code>)</td>
</tr>
<tr>
<td><strong>SerializeElement</strong></td>
<td>Writes the configuration data to an XML element when overridden in a derived class. (Inherited from <code>ConfigurationElement</code>)</td>
</tr>
<tr>
<td><strong>SerializeToXmlElement</strong></td>
<td>Writes the outer tags of this configuration element when implemented in a derived class. (Inherited from <code>ConfigurationElement</code>)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SetPropertyValue</td>
<td>Sets a property to the specified value. (Inherited from <code>ConfigurationElement</code>)</td>
</tr>
<tr>
<td>SetReadOnly</td>
<td>Sets the <code>IsReadOnly()</code> property for the object and for all sub-elements.   (Inherited from <code>ConfigurationElement</code>)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <code>String</code> that represents the object. (Inherited from <code>Object</code>)</td>
</tr>
<tr>
<td>Unmerge</td>
<td>Reverses the effect of merging configuration information from different levels of the configuration hierarchy. (Inherited from <code>ConfigurationElement</code>)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AddElementName</strong></td>
<td>Gets or sets the name of the <a href="ConfigurationElement">ConfigurationElement</a> to associate with the add operation in the <strong>ConfigurationElementCollection</strong> when overridden in a derived class. (Inherited from <strong>ConfigurationElementCollection</strong>)</td>
</tr>
<tr>
<td><strong>ClearElementName</strong></td>
<td>Gets or sets the name for the <a href="ConfigurationElement">ConfigurationElement</a> to associate with the clear operation in the <strong>ConfigurationElementCollection</strong> when overridden in a derived class. (Inherited from <strong>ConfigurationElementCollection</strong>)</td>
</tr>
<tr>
<td><strong>CollectionType</strong></td>
<td>Gets the type of the <a href="ConfigurationElementCollection">ConfigurationElementCollection</a>. (Overrides <strong>ConfigurationElementCollection.CollectionType</strong>)</td>
</tr>
<tr>
<td><strong>Count</strong></td>
<td>Gets the number of elements in the collection. (Inherited from <strong>ConfigurationElementCollection</strong>)</td>
</tr>
<tr>
<td><strong>CurrentConfiguration</strong></td>
<td>Gets a reference to the top-level <a href="Configuration">Configuration</a> instance that represents the configuration hierarchy that the current <a href="ConfigurationElement">ConfigurationElement</a> instance belongs to. (Inherited from <strong>ConfigurationElement</strong>.)</td>
</tr>
<tr>
<td><strong>ElementInformation</strong></td>
<td>Gets an <a href="ElementInformation">ElementInformation</a> object that contains the non-customizable information and functionality of the <a href="ConfigurationElement">ConfigurationElement</a> object. (Inherited from <strong>ConfigurationElement</strong>.)</td>
</tr>
<tr>
<td><strong>ElementName</strong></td>
<td>Gets the name used to identify this collection of elements. (Overrides <strong>ConfigurationElementCollection,ElementName</strong>)</td>
</tr>
<tr>
<td><strong>ElementProperty</strong></td>
<td>Gets the <a href="ConfigurationElementProperty">ConfigurationElementProperty</a> object that represents the <a href="ConfigurationElement">ConfigurationElement</a> object itself. (Inherited from <strong>ConfigurationElement</strong>.)</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>EmitClear</td>
<td>Gets or sets a value that specifies whether the collection has been cleared. (Inherited from ConfigurationElementCollection)</td>
</tr>
<tr>
<td>EvaluationContext</td>
<td>Gets the ContextInformation object for the ConfigurationElement object. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>IsSynchronized</td>
<td>Gets a value indicating whether access to the collection is synchronized (thread safe). (Inherited from ConfigurationElementCollection)</td>
</tr>
<tr>
<td>Item(ConfigurationProperty)</td>
<td>Gets or sets a property or attribute of this configuration element. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>Item(String)</td>
<td>Gets or sets a property, attribute, or child element of this configuration element. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>Item(Int32)</td>
<td>Gets the SolverElement at the specified index.</td>
</tr>
<tr>
<td>Item(Object)</td>
<td>Gets the SolverElement with the specified key.</td>
</tr>
<tr>
<td>LockAllAttributesExcept</td>
<td>Gets the collection of locked attributes. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>LockAllElementsExcept</td>
<td>Gets the collection of locked elements. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>LockAttributes</td>
<td>Gets the collection of locked attributes (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>LockElements</td>
<td>Gets the collection of locked elements. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>LockItem</td>
<td>Gets or sets a value indicating whether the element is locked. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>Properties</td>
<td>Gets the collection of properties. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>RemoveElementName</td>
<td>Gets or sets the name of the ConfigurationElement to associate with the remove operation in the ConfigurationElementCollection when overridden in a derived class. (Inherited from ConfigurationElementCollection)</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>SyncRoot</strong></td>
<td>Gets an object used to synchronize access to the <code>ConfigurationElementCollection</code>. (Inherited from <code>ConfigurationElementCollection</code>)</td>
</tr>
<tr>
<td><strong>ThrowOnDuplicate</strong></td>
<td>Gets a value indicating whether an attempt to add a duplicate <code>ConfigurationElement</code> to the <code>ConfigurationElementCollection</code> will cause an exception to be thrown. (Inherited from <code>ConfigurationElementCollection</code>)</td>
</tr>
</tbody>
</table>
See Also

SolversCollection Class
Optimization.Configuration Namespace
Initialization a new instance of the SolversCollection class

**Namespace:** Optimization.Configuration

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>public SolversCollection()</code></td>
<td><code>Public Sub New</code></td>
<td><code>public: SolversCollection()</code></td>
</tr>
</tbody>
</table>
See Also

SolversCollection Class
Optimization.Configuration Namespace
The `SolversCollection` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Adds the specified <a href="#">SolverElement</a> to the collection.</td>
</tr>
<tr>
<td>BaseAdd(ConfigurationElement)</td>
<td>Adds a configuration element to the collection. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td>BaseAdd(ConfigurationElement, Boolean)</td>
<td>Adds a configuration element to the collection. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td>BaseAdd(Int32, ConfigurationElement)</td>
<td>Adds a configuration element to the collection. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td>BaseClear</td>
<td>Removes all configuration element objects from the collection. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td>BaseGet(Object)</td>
<td>Returns the configuration element with the specified key. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td>BaseGet(Int32)</td>
<td>Gets the configuration element at the specified index. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td>BaseGetAllKeys</td>
<td>Returns an array of the keys for all of the configuration elements contained in the <a href="#">ConfigurationElementCollection</a>. (Inherited from <a href="#">ConfigurationElementCollection</a>)</td>
</tr>
<tr>
<td>BaseGetKey</td>
<td>Gets the key for the <a href="#">ConfigurationElement</a>. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td>BaseIndexOf</td>
<td>The index of the specified <a href="#">ConfigurationElement</a>. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td>BaseIsRemoved</td>
<td>Gets a value indicating whether the specified key has been removed from the <a href="#">ConfigurationElementCollection</a>. (Inherited from <a href="#">ConfigurationElementCollection</a>)</td>
</tr>
<tr>
<td>BaseRemove</td>
<td>Removes a <a href="#">ConfigurationElement</a> from the collection. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td>BaseRemoveAt</td>
<td>Removes the <a href="#">ConfigurationElement</a> at the specified index. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td>CopyTo</td>
<td>Copies the contents of the <a href="#">ConfigurationElement</a>. (Inherited from <a href="#">ConfigurationElement</a>)</td>
</tr>
<tr>
<td>CreateNewElement()</td>
<td>Creates a new <a href="#">SolverElement</a>. (Overrides <a href="#">ConfigurationElementCollection</a>)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>CreateNewElement(String)</td>
<td>Creates a new ConfigurationElement (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>DeserializeElement</td>
<td>Reads XML from the configuration file. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>Equals</td>
<td>Compares the ConfigurationElement (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free operations before the Object is reclaimed. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetElementKey</td>
<td>Gets the element key for the specified configuration element. (Overrides ConfigurationElementCollection.GetElementKey(ConfigurationElement))</td>
</tr>
<tr>
<td>GetEnumerator</td>
<td>Gets an IEnumerator which is used to iterate through the ConfigurationElementCollection. (Inherited from ConfigurationElementCollection)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Gets a unique value representing the instance. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>GetItemAt</td>
<td>Gets the SolverElement at the specified index.</td>
</tr>
<tr>
<td>GetItemByKey</td>
<td>Gets the SolverElement with the specified key.</td>
</tr>
<tr>
<td>GetTransformedAssemblyString</td>
<td>Returns the transformed version of the specified assembly name. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>GetTransformedTypeString</td>
<td>Returns the transformed version of the specified type name. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td>Init</td>
<td>Sets the ConfigurationElement object to its initial state. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>InitializeDefault</td>
<td>Used to initialize a default set of values for the object. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>IsElementName</td>
<td>Indicates whether the specified ConfigurationElementCollection. (Overrides ConfigurationElementCollection)</td>
</tr>
<tr>
<td>IsElementRemovable</td>
<td>Gets a value indicating whether the</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>IsModified</td>
<td>Indicates whether this ConfigurationElementCollection since it was last saved or loaded when overridden in a derived class.</td>
</tr>
<tr>
<td>IsReadOnly</td>
<td>Gets a value indicating whether the element is read-only. (Overrides ConfigurationElementCollection.IsReadOnly)</td>
</tr>
<tr>
<td>ListErrors</td>
<td>Adds the invalid-property errors in this element and in all subelements, to the passed list. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current ConfigurationElementCollection. (Inherited from Object.)</td>
</tr>
<tr>
<td>OnDeserializeUnrecognizedAttribute</td>
<td>Gets a value indicating whether an unknown attribute is encountered during deserialization. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>OnDeserializeUnrecognizedElement</td>
<td>Causes the configuration system to throw an exception. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>OnRequiredPropertyNotFound</td>
<td>Throws an exception when a required property is not found. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>PostDeserialize</td>
<td>Called after deserialization. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>PreSerialize</td>
<td>Called before serialization. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>Remove</td>
<td>Removes the specified SolverElement ConfigurationElementCollection.</td>
</tr>
<tr>
<td>Reset</td>
<td>Resets the ConfigurationElementCollection in a derived class. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>ResetModified</td>
<td>Resets the value of the IsModified() derived class. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>SerializeElement</td>
<td>Writes the configuration data to an XDocument when overridden in a derived class. (Inherited from ConfigurationElement)</td>
</tr>
<tr>
<td>SerializeToXmlElement</td>
<td>Writes the outer tags of this configuration element when implemented in a derived class.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SetPropertyVal</td>
<td>Sets a property to the specified value.</td>
</tr>
<tr>
<td>SetReadOnly</td>
<td>Sets the <code>IsReadOnly()</code> property for the object and for all sub-elements.</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <code>String</code> that represents the object.</td>
</tr>
<tr>
<td>Unmerge</td>
<td>Reverses the effect of merging configuration information from different</td>
</tr>
<tr>
<td></td>
<td>levels of the configuration hierarchy.</td>
</tr>
</tbody>
</table>
See Also

SolversCollection Class
Optimization.Configuration Namespace
SolversCollection.Add Method

Optimization.Configuration.SolversCollection Class

Adds the specified SolverElement to the ConfigurationElementCollection.

Namespace: Optimization.Configuration
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public void Add(&lt;br&gt;  SolverElement solver)</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Sub Add ( solved As SolverElement _ )</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public:&lt;br&gt; void Add(&lt;br&gt; SolverElement^ solver)</td>
</tr>
</tbody>
</table>

### Parameters

- **solver**
  
  Type: `Optimization.Configuration.SolverElement`

  The `SolverElement` to add.
See Also

SolversCollection Class
Optimization.Configuration Namespace
Optimization Framework

SolversCollection.BaseAdd Method

SolversCollection Class  See Also  Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BaseAdd(ConfigurationElement)</td>
<td>Adds a configuration element to the ConfigurationElementCollection. (Inherited from ConfigurationElementCollection.)</td>
</tr>
<tr>
<td>BaseAdd(ConfigurationElement, Boolean)</td>
<td>Adds a configuration element to the configuration element collection. (Inherited from ConfigurationElementCollection.)</td>
</tr>
<tr>
<td>BaseAdd(Int32, ConfigurationElement)</td>
<td>Adds a configuration element to the configuration element collection. (Inherited from ConfigurationElementCollection.)</td>
</tr>
</tbody>
</table>
See Also

SolversCollection Class
SolversCollection Members
Optimization.Configuration Namespace
Optimization Framework

**SolversCollection.BaseGet Method**

[SolversCollection Class] See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BaseGet(Object)</td>
<td>Returns the configuration element with the specified key. (Inherited from <a href="#">ConfigurationElementCollection</a>.)</td>
</tr>
<tr>
<td>BaseGet(Int32)</td>
<td>Gets the configuration element at the specified index location. (Inherited from <a href="#">ConfigurationElementCollection</a>.)</td>
</tr>
</tbody>
</table>
See Also

SolversCollection Class
SolversCollection Members
Optimization.Configuration Namespace
Optimization Framework

**SolversCollection.CreateNewElement Method**

[SolversCollection Class](#)  [See Also](#)  [Send Feedback](#)
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CreateNewElement()</code></td>
<td>Creates a new <code>SolverElement</code>. (Overrides <code>ConfigurationElementCollection.CreateNewElement</code>.)</td>
</tr>
<tr>
<td><code>CreateNewElement(String)</code></td>
<td>Creates a new <code>ConfigurationElement</code> when overridden in a derived class. (Inherited from <code>ConfigurationElementCollection</code>.)</td>
</tr>
</tbody>
</table>
See Also

SolversCollection Class
SolversCollection Members
Optimization.Configuration Namespace
SolversCollection.CreateNewElement Method

Creates a new SolverElement.

Namespace: Optimization.Configuration
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>protected override ConfigurationElement CreateNewElement</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Protected Overrides Function CreateNewElement As ConfigurationElement</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>protected: virtual ConfigurationElement^ CreateNewElement() override</code></td>
</tr>
</tbody>
</table>

### Return Value

A new `SolverElement`. 
## Contracts

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Contract.Result&lt;ConfigurationElement&gt;() != null</code></td>
</tr>
</tbody>
</table>

*Inherited from:* [ConfigurationElementCollection](#)

[Learn more about contracts](#)
See Also

SolversCollection Class
CreateNewElement Overload
Optimization.Configuration Namespace
Optimization Framework

SolversCollection.GetElementKey Method

Gets the element key for the specified configuration element.

Namespace: Optimization.Configuration
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
protected override Object GetElementKey(ConfigurationElement element)
```

**Visual Basic**

```vbnet
Protected Overrides Function GetElementKey ( _
    element As ConfigurationElement _
) As Object
```

**Visual C++**

```cpp
protected:
virtual Object^ GetElementKey(
    ConfigurationElement^ element
) override
```

### Parameters

**element**

Type: `System.Configuration.ConfigurationElement`

The `ConfigurationElement` to return the key for.

### Return Value

An `Object` that acts as the key for the specified `ConfigurationElement`. 
### Contracts

#### Requires

<table>
<thead>
<tr>
<th>element</th>
<th>!=</th>
<th>null</th>
</tr>
</thead>
</table>

*Inherited From:* [ConfigurationElementCollection](#)  

#### Ensures

<table>
<thead>
<tr>
<th>Contract.Result&lt;object&gt;()</th>
<th>!=</th>
<th>null</th>
</tr>
</thead>
</table>

*Inherited From:* [ConfigurationElementCollection](#)  

[Learn more about contracts](#)
See Also

SolversCollection Class
Optimization.Configuration Namespace
Gets the `SolverElement` at the specified index.

**Namespace:** [Optimization.Configuration](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public SolverElement GetItemAt(
    int index
)
```

**Visual Basic**

```vbnet
Public Function GetItemAt (_
    index As Integer _
) As SolverElement
```

**Visual C++**

```cpp
public:
SolverElement^ GetItemAt(
    int index
)
```

### Parameters

- **index**
  - Type: `System.Int32`
  - The index of the `SolverElement` to retrieve.

### Return Value

See Also

SolversCollection Class
Optimization.Configuration Namespace
SolversCollection.GetItemByKey Method

Gets the SolverElement with the specified key.

Namespace: Optimization.Configuration
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
# Syntax

## C#

```csharp
public SolverElement GetItemByKey(
    string name
)
```

## Visual Basic

```vb
Public Function GetItemByKey (  
    name As String  
) As SolverElement
```

## Visual C++

```cpp
public:
SolverElement^ GetItemByKey(
    String^ name
)
```

## Parameters

*name*

Type: `System.String`

The key of the `SolverElement` to retrieve.

## Return Value


[... ]
See Also

SolversCollection Class
Optimization.Configuration Namespace
Indicates whether the specified ConfigurationElement exists in the ConfigurationElementCollection.

Namespace: Optimization.Configuration
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
protected override bool IsElementName(
    string elementName
)
```

### Visual Basic

```vbnet
Protected Overrides Function IsElementName ( _
    elementName As String _
) As Boolean
```

### Visual C++

```cpp
protected:
virtual bool IsElementName(
    String^ elementName
) override
```

## Parameters

`elementName`

Type: `System.String`

The name of the element to verify.

## Return Value

true (True in Visual Basic) if the element exists in the collection; otherwise, false (False in Visual Basic).
See Also

SolversCollection Class
Optimization.Configuration Namespace
Gets a value indicating whether the element is read-only.

**Namespace:** Optimization.Configuration  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public override bool IsReadOnly()</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Overrides Function IsReadOnly As Boolean</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public: virtual bool IsReadOnly() override</code></td>
</tr>
</tbody>
</table>

### Return Value

See Also

SolversCollection Class
Optimization.Configuration Namespace
Removes the specified **SolverElement** from the **ConfigurationElementCollection**.

**Namespace:** Optimization.Configuration  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#**  | ```
public void Remove(
               SolverElement solver
)
``` |
| **Visual Basic** | ```
Public Sub Remove (_
               solver As SolverElement _
)
``` |
| **Visual C++** | ```
public:
   void Remove(
               SolverElement^ solver
)
``` |

### Parameters

- **solver**
  - Type: **Optimization.Configuration.SolverElement**
  - The **SolverElement** to remove.
Contracts

**Requires**

`solver!=null`

[Learn more about contracts](#)
See Also

SolversCollection Class
Optimization.Configuration Namespace
The **SolversCollection** type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AddElementName</strong></td>
<td>Gets or sets the name of the <a href="#">ConfigurationElement</a> to associate with the add operation in the <a href="#">ConfigurationElementCollection</a> when overridden in a derived class. (Inherited from <a href="#">ConfigurationElementCollection</a>)</td>
</tr>
<tr>
<td><strong>ClearElementName</strong></td>
<td>Gets or sets the name for the <a href="#">ConfigurationElement</a> to associate with the clear operation in the <a href="#">ConfigurationElementCollection</a> when overridden in a derived class. (Inherited from <a href="#">ConfigurationElementCollection</a>)</td>
</tr>
<tr>
<td><strong>CollectionType</strong></td>
<td>Gets the type of the <a href="#">ConfigurationElementCollection</a>. (Overrides <a href="#">ConfigurationElementCollection</a>.)</td>
</tr>
<tr>
<td><strong>Count</strong></td>
<td>Gets the number of elements in the collection. (Inherited from <a href="#">ConfigurationElementCollection</a>.)</td>
</tr>
<tr>
<td><strong>CurrentConfiguration</strong></td>
<td>Gets a reference to the top-level <a href="#">Configuration</a> instance that represents the configuration hierarchy that the current <a href="#">ConfigurationElement</a> instance belongs to. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
<tr>
<td><strong>ElementInformation</strong></td>
<td>Gets an <a href="#">ElementInformation</a> object that contains the non-customizable information and functionality of the <a href="#">ConfigurationElement</a> object. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
<tr>
<td><strong>ElementName</strong></td>
<td>Gets the name used to identify this collection of elements (Overrides <a href="#">ConfigurationElementCollection,ElementName</a>.)</td>
</tr>
<tr>
<td><strong>ElementProperty</strong></td>
<td>Gets the <a href="#">ConfigurationElementProperty</a> object that represents the <a href="#">ConfigurationElement</a> object itself. (Inherited from <a href="#">ConfigurationElement</a>.)</td>
</tr>
<tr>
<td><strong>EmitClear</strong></td>
<td>Gets or sets a value that specifies whether the collection has been cleared. (Inherited from <code>ConfigurationElementCollection</code>)</td>
</tr>
<tr>
<td><strong>EvaluationContext</strong></td>
<td>Gets the <code>ContextInformation</code> object for the <code>ConfigurationElement</code> object. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>IsSynchronized</strong></td>
<td>Gets a value indicating whether access to the collection is synchronized (thread safe). (Inherited from <code>ConfigurationElementCollection</code>.)</td>
</tr>
<tr>
<td><strong>Item(ConfigurationProperty)</strong></td>
<td>Gets or sets a property or attribute of this configuration element. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>Item(String)</strong></td>
<td>Gets or sets a property, attribute, or child element of this configuration element. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>Item(Int32)</strong></td>
<td>Gets the <code>SolverElement</code> at the specified index.</td>
</tr>
<tr>
<td><strong>Item(Object)</strong></td>
<td>Gets the <code>SolverElement</code> with the specified key.</td>
</tr>
<tr>
<td><strong>LockAllAttributesExcept</strong></td>
<td>Gets the collection of locked attributes. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>LockAllElementsExcept</strong></td>
<td>Gets the collection of locked elements. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>LockAttributes</strong></td>
<td>Gets the collection of locked attributes. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>LockElements</strong></td>
<td>Gets the collection of locked elements. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>LockItem</strong></td>
<td>Gets or sets a value indicating whether the element is locked. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>Properties</strong></td>
<td>Gets the collection of properties. (Inherited from <code>ConfigurationElement</code>.)</td>
</tr>
<tr>
<td><strong>RemoveElementName</strong></td>
<td>Gets or sets the name of the <code>ConfigurationElement</code> to associate with the remove operation in the <code>ConfigurationElementCollection</code> when overridden in a derived class. (Inherited from <code>ConfigurationElementCollection</code>.)</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>SyncRoot</strong></td>
<td>Gets an object used to synchronize access to the <a href="#">ConfigurationElementCollection</a>. (Inherited from <a href="#">ConfigurationElementCollection</a>)</td>
</tr>
<tr>
<td><strong>ThrowOnDuplicate</strong></td>
<td>Gets a value indicating whether an attempt to add a duplicate <a href="#">ConfigurationElement</a> to the <a href="#">ConfigurationElementCollection</a> will cause an exception to be thrown. (Inherited from <a href="#">ConfigurationElementCollection</a>)</td>
</tr>
</tbody>
</table>
See Also

SolversCollection Class
Optimization.Configuration Namespace
SolversCollection.CollectionType Property

Gets the type of the `ConfigurationElementCollection`.

Namespace: `Optimization.Configuration`
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public override <a href="#">ConfigurationElementCollectionType</a> CollectionType</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Overrides ReadOnly Property CollectionType As Get</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: virtual property <a href="#">ConfigurationElementCollectionType</a> CollectionType get () override</td>
</tr>
</tbody>
</table>

### Return Value

The [ConfigurationElementCollectionType](#) of this collection.
See Also

SolversCollection Class
Optimization.Configuration Namespace
Optimization Framework

SolversCollection.ElementName Property

Namespace: Optimization.Configuration
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0

Gets the name used to identify this collection of elements.
### Syntax

#### C#

```csharp
protected override string ElementName { get; }
```

#### Visual Basic

```vbnet
Protected Overrides ReadOnly Property ElementName As Get
```

#### Visual C++

```cpp
protected:
virtual property String^ ElementName {
    String^ get () override;
}
```
Contracts

Get

<table>
<thead>
<tr>
<th>Ensures</th>
</tr>
</thead>
</table>

Contract.Result<string>(() != null

Inherited ConfigurationElementCollection

From:

Learn more about contracts
See Also

SolversCollection Class
Optimization.Configuration Namespace
Optimization Framework

SolversCollection.Item Property

SolversCollection Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item(ConfigurationProperty)</td>
<td>Gets or sets a property or attribute of this configuration element. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Item(String)</td>
<td>Gets or sets a property, attribute, or child element of this configuration element. (Inherited from ConfigurationElement.)</td>
</tr>
<tr>
<td>Item(Int32)</td>
<td>Gets the SolverElement at the specified index.</td>
</tr>
<tr>
<td>Item(Object)</td>
<td>Gets the SolverElement with the specified key.</td>
</tr>
</tbody>
</table>
See Also

- SolversCollection Class
- SolversCollection Members
- Optimization.Configuration Namespace
SolversCollection.Item Property (Int32)

SolversCollection Class  See Also  Send Feedback

Gets the **SolverElement** at the specified index.

**Namespace:** Optimization.Configuration  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#**     | ```csharp
public SolverElement this[int index]
    {
    get;
    }
``` |
| **Visual Basic** | `Public ReadOnly Default Property Item ( _
          index As Integer _
       ) As SolverElement
              Get` |
| **Visual C++** | ```cpp
public:
    property SolverElement^ default[int index] {
        SolverElement^ get (int index);
    }
``` |

### Parameters

- **index**
  - Type: `System.Int32`
  - The index of the `SolverElement` to retrieve.
See Also

SolversCollection Class
Item Overload
Optimization.Configuration Namespace
Gets the SolverElement with the specified key.

**Namespace:** Optimization.Configuration

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public SolverElement this[Object name] { get; }
```

**Visual Basic**

```vbnet
Public ReadOnly Default Property Item (name As Object) As SolverElement
    Get
```

**Visual C++**

```cpp
public:
property SolverElement^ default[Object^ name] { SolverElement^ get (Object^ name); }
```

### Parameters

**name**

Type: `System.Object`

The key of the `SolverElement` to retrieve.
See Also

SolversCollection Class
Item Overload
Optimization.Configuration Namespace
Optimization Framework

Optimization.Exporter Namespace

[Missing <summary> documentation for "N:Optimization.Exporter"]
### Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExpressionNormalizer</td>
<td>Class used for normalizing an expression. ConstantExpressions should be added up or expanded (to show if the expression is linear or quadratic or nonlinear) Another goal is to eliminate as many ConstantExpressions as possible.</td>
</tr>
<tr>
<td>LPExporter</td>
<td>This class can export a model into a stream, using the LP file format.</td>
</tr>
<tr>
<td>MPSExporter</td>
<td>This class can export a model into a stream, using the MPS file format.</td>
</tr>
</tbody>
</table>
Class used for normalizing an expression. ConstantExpressions should be added up or expanded (to show if the expression is linear or quadratic or nonlinear) Another goal is to eliminate as many ConstantExpressions as possible.

**Namespace:** Optimization.Exporter  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
<tr>
<td><code>public class ExpressionNormalizer : IExpressionVisitor</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td></td>
</tr>
<tr>
<td><code>Public Class ExpressionNormalizer _ Implements IExpressionVisitor(Of Expression)</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
</tr>
<tr>
<td><code>public ref class ExpressionNormalizer : IExpressionVisitor</code></td>
<td></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

- `System.Object`
- `Optimization.Exporter.ExpressionNormalizer`
See Also

ExpressionNormalizer Members
Optimization.Exporter Namespace
The **ExpressionNormalizer** type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExpressionNormalizer</td>
<td>Initializes a new instance of the ExpressionNormalizer class</td>
</tr>
</tbody>
</table>
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is equal to the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.type">Type</a> of the current instance. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="https://docs.microsoft.com/en-us/dotnet/api/system.string">String</a> that represents the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Visit</strong></td>
<td>Returns the normalized expression</td>
</tr>
</tbody>
</table>
See Also

ExpressionNormalizer Class
Optimization.Exporter Namespace
Initializes a new instance of the `ExpressionNormalizer` class

**Namespace:** [Optimization.Exporter](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code Sample</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public ExpressionNormalizer()</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Sub New</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public: ExpressionNormalizer()</code></td>
</tr>
</tbody>
</table>
See Also

ExpressionNormalizer Class
Optimization.Exporter Namespace
The `ExpressionNormalizer` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is equal to the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.type">Type</a> of the current instance. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="https://docs.microsoft.com/en-us/dotnet/api/system.string">String</a> that represents the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Visit</strong></td>
<td>Returns the normalized expression</td>
</tr>
</tbody>
</table>
See Also

ExpressionNormalizer Class
Optimization.Exporter Namespace
Returns the normalized expression

Namespace: Optimization.Exporter
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>public Expression Visit( Expression expression )</td>
</tr>
<tr>
<td>Visual Basic</td>
<td>Public Function Visit ( _expression As Expression _ ) As Expression</td>
</tr>
<tr>
<td>Visual C++</td>
<td>public: virtual Expression^ Visit( Expression^ expression ) sealed</td>
</tr>
</tbody>
</table>

### Parameters

- **expression**
  - Type: Optimization.Expression
  - The Optimization.Model.Expression

### Return Value

- The normalized expression
Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>(object) exp! = null</td>
</tr>
</tbody>
</table>

Inherited from: IExpressionVisitor

Learn more about contracts
See Also

ExpressionNormalizer Class
Optimization.Exporter Namespace
This class can export a model into a stream, using the LP file format.

**Namespace:** [Optimization.Exporter](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public class LPExporter</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Class LPExporter</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public ref class LPExporter</code></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
Optimization.Exporter.LPExporter
See Also

LPExporter Members
OptimizationExporter Namespace
The **LPExporter** type exposes the following members.
### Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPExporter</td>
<td>Creates a new instance of class LPExporter.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is equal to the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.type">Type</a> of the current instance. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="https://docs.microsoft.com/en-us/dotnet/api/system.string">String</a> that represents the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Write</strong></td>
<td>Delegates to a method for each part of a LP file.</td>
</tr>
</tbody>
</table>
See Also

LPExporter Class
Optimization.Exporter Namespace
Optimization Framework

LPExporter Constructor

LPExporter Class

See Also

Send Feedback

Creates a new instance of class LPExporter.

Namespace: Optimization.Exporter

Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
</table>
| public LPExporter(
    Stream filestream
) | Public Sub New (_
    filestream As Stream _
) | public:
    LPExporter(
        Stream^ filestream
    ) |

**Parameters**

`filestream`

Type: `System.IO.Stream`  
The stream in which the model should be written.
See Also

LPExporter Class
Optimization.Exporter Namespace
The **LPExporter** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Write</strong></td>
<td>Delegates to a method for each part of a LP file.</td>
</tr>
</tbody>
</table>
See Also

LPExporter Class
Optimization.Exporter Namespace
Delegates to a method for each part of a LP file.

**Namespace:** Optimization.Exporter  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#**    | ```
public void Write(
    Model model
)
``` |
| **Visual Basic** | ```
Public Sub Write ( ___
    model As Model ___
)
``` |
| **Visual C++** | ```
public:
void Write(
    Model^ model
)
``` |

### Parameters

- **model**
  - Type: [Optimization.Model](#)
  - The model that should be exported.
See Also

LPExporter Class
Optimization.Exporter Namespace
This class can export a model into a stream, using the MPS file format.

**Namespace:** [Optimization.Exporter](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>public class MPSExporter</td>
</tr>
<tr>
<td>Visual Basic</td>
<td>Public Class MPSExporter</td>
</tr>
<tr>
<td>Visual C++</td>
<td>public ref class MPSExporter</td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object

Optimization.Exporter.MPSExporter
See Also

MPSExporter Members
Optimization.Exporter Namespace
The **MPSExporter** type exposes the following members.
### Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPSExporter</td>
<td>Creates a new instance of class MPSExporter.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="/Object">Object</a> is equal to the current <a href="/Object">Object</a>. (Inherited from <a href="/Object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="/Object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="/Object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="/Object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="/Object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="/Type">Type</a> of the current instance. (Inherited from <a href="/Object">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="/Object">Object</a>. (Inherited from <a href="/Object">Object</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="/String">String</a> that represents the current <a href="/Object">Object</a>. (Inherited from <a href="/Object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Write</strong></td>
<td>Delegates to a method for each section of a MPS file.</td>
</tr>
</tbody>
</table>
See Also

MPSExporter Class
Optimization.Exporter Namespace
Optimization Framework

**MPSExporter Constructor**

See Also: MPSExporter Class

---

Creates a new instance of class **MPSExporter**.

**Namespace**: Optimization.Exporter

**Assembly**: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public MPSExporter(</td>
</tr>
<tr>
<td></td>
<td>Stream filestream</td>
</tr>
<tr>
<td></td>
<td>)</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Sub New ( _</td>
</tr>
<tr>
<td></td>
<td>filestream As Stream _ )</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public:</td>
</tr>
<tr>
<td></td>
<td>MPSExporter(</td>
</tr>
<tr>
<td></td>
<td>Stream^ filestream</td>
</tr>
<tr>
<td></td>
<td>)</td>
</tr>
</tbody>
</table>

### Parameters

- **filestream**
  - Type: `System.IO.Stream`
  - The stream in which the model should be written.
See Also

MPSExporter Class

Optimization.Exporter Namespace
Optimization Framework

MPSExporter Methods

MPSExporter Class See Also Send Feedback

The MPSExporter type exposes the following members.
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Write</strong></td>
<td>Delegates to a method for each section of a MPS file.</td>
</tr>
</tbody>
</table>
See Also

MPSExporter Class
Optimization.Exporter Namespace
Delegates to a method for each section of a MPS file.

**Namespace:** Optimization.Exporter

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>public void Write(</td>
<td></td>
</tr>
<tr>
<td>Model model</td>
<td></td>
</tr>
<tr>
<td>)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual Basic</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Sub Write (</td>
<td></td>
</tr>
<tr>
<td>model As Model _</td>
<td></td>
</tr>
<tr>
<td>)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual C++</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>public:</td>
<td></td>
</tr>
<tr>
<td>void Write(</td>
<td></td>
</tr>
<tr>
<td>Model^ model</td>
<td></td>
</tr>
<tr>
<td>)</td>
<td></td>
</tr>
</tbody>
</table>

### Parameters

- **model**
  - Type: [Optimization,Model]
  - The model that should be exported.
See Also

MPSExporter Class
Optimization.Exporter Namespace
Holds the main interfaces of the framework
## Interfaces

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICanTransform(M1, M2)</td>
<td>A class which implements this interface can convert objects of type M1 to objects of type M2</td>
</tr>
<tr>
<td>ISetAccessor(T, S)</td>
<td>Defines a Set and an Accessor function</td>
</tr>
<tr>
<td>ISolver</td>
<td>Represents a solver instance for mathematical programming problems.</td>
</tr>
<tr>
<td>IVariable</td>
<td>Represents a variable in an IModel.</td>
</tr>
</tbody>
</table>
A class which implements this interface can convert objects of type M1 to objects of type M2

**Namespace:** Optimization.Interfaces  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public interface ICanTransform&lt;M1, M2&gt;</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Interface ICanTransform(Of M1, M2)</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>generic&lt;typename M1, typename M2&gt;</code>&lt;br&gt;<code>public interface class ICanTransform</code></td>
</tr>
</tbody>
</table>
Type Parameters

\( M1 \)
Type to convert from

\( M2 \)
Type to convert to
See Also

ICanTransform(M1, M2) Members
Optimization.Interfaces Namespace
The `ICanTransform(M1, M2)` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transform</td>
<td></td>
</tr>
</tbody>
</table>
See Also

ICanTransform(M1, M2) Interface
Optimization.Interfaces Namespace
The ICanTransform(M1, M2) type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transform</td>
<td></td>
</tr>
</tbody>
</table>
See Also

ICanTransform(M1, M2) Interface
Optimization.Interfaces Namespace
ICanTransform(\(M_1, M_2\)).Transform Method

ICanTransform(\(M_1, M_2\)) Interface  See Also  Send Feedback

[Missing <summary> documentation for "M:Optimization.Interfaces.ICanTransform`2.Transform`0"]

Namespace: Optimization.Interfaces
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
M2 Transform(
    M1 model
)
```

#### Visual Basic

```vbnet
Function Transform (_
    model As M1 _
) As M2
```

#### Visual C++

```cpp
M2 Transform(
    M1 model
)
```

### Parameters

- **model**
  - Type: `M1`
  - [Missing <param name="model"/> documentation for "M:Optimization.Interfaces.ICanTransform`2.Transform(`0)"]

### Return Value

- [Missing <returns> documentation for "M:Optimization.Interfaces.ICanTransform`2.Transform(`0)"]
See Also

ICanTransform(M1, M2) Interface
Optimization.Interfaces Namespace
Defines a Set and an Accessor function

**Namespace:** [Optimization.Interfaces](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
</table>
|                 | public interface ISetAccessor<T, out S> | Public Interface ISetAccessor(Of T, Out S) | generic< typename T, typename S >
|                 |                                        |                                | public interface class ISetAccessor |
Type Parameters

$T$
The type of the Set

$S$
The return type of the Accessor function
See Also

ISetAccessor(T, S) Members
Optimization.Interfaces Namespace
Optimization Framework

**ISetAccessor**\((T, S)\) Members

- **ISetAccessor**\((T, S)\) Interface Properties See Also Send Feedback

The **ISetAccessor**\((T, S)\) type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accessor</td>
<td>Gets the accessor.</td>
</tr>
<tr>
<td>Set</td>
<td>Gets the set.</td>
</tr>
</tbody>
</table>
See Also

ISetAccessor(T, S) Interface
Optimization.Interfaces Namespace
The `ISetAccessor(T, S)` type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accessor</td>
<td>Gets the accessor.</td>
</tr>
<tr>
<td>Set</td>
<td>Gets the set.</td>
</tr>
</tbody>
</table>
See Also

ISetAccessor(T, S) Interface
Optimization.Interfaces Namespace
Gets the accessor.

**Namespace:** Optimization.Interfaces

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
</table>
| `Func<T, S>Accessor { get; }` | `ReadOnly Property Accessor As Func(Of T, S) Get` | `property Func<T, S>^Accessor {
  Func<T, S>^ get();
}
` |

### Field Value

The accessor.
See Also

ISetAccessor(T, S) Interface
Optimization.Interfaces Namespace
Gets the set.

**Namespace:** Optimization.Interfaces  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>IEnumerable&lt;S&gt; Set { get; }</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>ReadOnly Property Set As IEnumerable(Of S) Get</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>property IEnumerable&lt;S&gt;^ Set { IEnumerable&lt;S&gt;^ get (); }</code></td>
</tr>
</tbody>
</table>

**Field Value**

The set.
See Also

ISetAccessor(T, S) Interface
Optimization.Interfaces Namespace
Represents a solver instance for mathematical programming problems.

**Namespace:** Optimization.Interfaces  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td><code>public interface ISolver : ICanSolve&lt;Model, Solution&gt;, IAbortable</code></td>
</tr>
</tbody>
</table>
| Visual Basic | `Public Interface ISolver _
Inherits ICanSolve(Of Model, Solution), IAbortable` |
| Visual C++  | `public interface class ISolver : ICanSolve<Model^, Solution>, IAbortable` |
See Also

ISolver Members
Optimization.Interfaces Namespace
The **ISolver** type exposes the following members.
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Abort</strong></td>
<td>If this solver instance is busy abort the run as soon as possible, or do nothing if this solver instance is not busy. (Inherited from IAbortable.)</td>
</tr>
<tr>
<td><strong>ClearLastModel</strong></td>
<td>Deletes the internal datastructures of this solver instance.</td>
</tr>
<tr>
<td><strong>Solve</strong></td>
<td>Solves the given model. Optimizes if model contains at least one objective. (Inherited from ICanSolve(M, S).)</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td>The configuration of this solver instance.</td>
</tr>
<tr>
<td>IsBusy</td>
<td>Is this solver instance busy?</td>
</tr>
</tbody>
</table>
See Also

ISolver Interface
Optimization.Interfaces Namespace
The **ISolver** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abort</td>
<td>If this solver instance is busy abort the run as soon as possible, or do nothing if this solver instance is not busy. (Inherited from IAbortable.)</td>
</tr>
<tr>
<td>ClearLastModel</td>
<td>Deletes the internal datastructures of this solver instance.</td>
</tr>
<tr>
<td>Solve</td>
<td>Solves the given model. Optimizes if model contains at least one objective. (Inherited from ICanSolve(M, S).)</td>
</tr>
</tbody>
</table>
See Also

ISolver Interface
Optimization.Interfaces Namespace
Optimization Framework

**ISolver.ClearLastModel Method**

*ISolver Interface*  *See Also*  *Send Feedback*

Deletes the internal datastructures of this solver instance.

**Namespace:** [Optimization.Interfaces](https://example.com)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  **Version:** 0.3.1036.0
### Syntax

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>void ClearLastModel()</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>Sub ClearLastModel</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>void ClearLastModel()</code></td>
</tr>
</tbody>
</table>
### Exceptions

<table>
<thead>
<tr>
<th>Exception</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>System.InvalidOperationException</code></td>
<td>If this solver instance is busy.</td>
</tr>
</tbody>
</table>
See Also

ISolver Interface
Optimization Interfaces Namespace
The ISolver type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td>The configuration of this solver instance.</td>
</tr>
<tr>
<td>IsBusy</td>
<td>Is this solver instance busy?</td>
</tr>
</tbody>
</table>
See Also

ISolver Interface
Optimization.Interfaces Namespace
The configuration of this solver instance.

**Namespace:** [Optimization.Interfaces](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>SolverConfiguration</code> Configuration { get; set; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Property Configuration As SolverConfiguration Get Set</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>property SolverConfiguration^ Configuration { SolverConfiguration^ get (); void set (SolverConfiguration^ value); }</code></td>
</tr>
</tbody>
</table>
See Also

ISolver Interface
Optimization.Interfaces Namespace
Is this solver instance busy?

**Namespace:** [Optimization.Interfaces](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
bool IsBusy { get; }
```

**Visual Basic**

```vbnet
ReadOnly Property IsBusy As Boolean
    Get
```

**Visual C++**

```cpp
property bool IsBusy {
    bool get ();
}
```
See Also

ISolver Interface
Optimization.Interfaces Namespace
Represents a variable in an **IModel**.

**Namespace:** [Optimization.Interfaces](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
</tr>
<tr>
<td>public interface IVariable</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
</tr>
<tr>
<td>Public Interface IVariable</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
</tr>
<tr>
<td>public interface class IVariable</td>
</tr>
</tbody>
</table>
See Also

IVariable Members
Optimization.Interfaces Namespace
The **IVariable** type exposes the following members.
Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LowerBound</td>
<td>Lower bound of this variable.</td>
</tr>
<tr>
<td>Name</td>
<td>Unique name of this variable.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of this variable.</td>
</tr>
<tr>
<td>UpperBound</td>
<td>Upper bound of this variable.</td>
</tr>
<tr>
<td>Value</td>
<td>Gets or sets the value for this variable. This field can be used to assign solution values from a solver to the variables in the original model.</td>
</tr>
</tbody>
</table>
See Also

IVariable Interface
Optimization.Interfaces Namespace
The **IVariable** type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LowerBound</strong></td>
<td>Lower bound of this variable.</td>
</tr>
<tr>
<td><strong>Name</strong></td>
<td>Unique name of this variable.</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Type of this variable.</td>
</tr>
<tr>
<td><strong>UpperBound</strong></td>
<td>Upper bound of this variable.</td>
</tr>
<tr>
<td><strong>Value</strong></td>
<td>Gets or sets the value for this variable. This field can be used to assign solution values from a solver to the variables in the original model.</td>
</tr>
</tbody>
</table>
See Also

IVariable Interface
Optimization.Interfaces Namespace
Lower bound of this variable.

**Namespace:** [Optimization.Interfaces](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>double LowerBound { get; set; }</code></td>
</tr>
</tbody>
</table>
| **Visual Basic** | `Property LowerBound As Double
Get
Set` |
| **Visual C++** | `property double LowerBound {
    double get ();
    void set (double value);` |
See Also

IVariable Interface
Optimization.Interfaces Namespace
IVariable.Name Property

Unique name of this variable.

**Namespace:** [Optimization.Interfaces](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>string Name { get; set; }</td>
<td>Property Name As String</td>
<td>property String^ Name {</td>
</tr>
<tr>
<td></td>
<td>Get</td>
<td>String^ get ();</td>
</tr>
<tr>
<td></td>
<td>Set</td>
<td>void set (String^ value);</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>
See Also

IVariable Interface

Optimization.Interfaces Namespace
Type of this variable.

**Namespace:** [Optimization.Interfaces](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
VariableType Type { get; set; }
```

#### Visual Basic

```vbnet
Property Type As VariableType
    Get
    Set
```

#### Visual C++

```cpp
property VariableType Type {
    VariableType get ();
    void set (VariableType value);
}
```
See Also

IVariable Interface
Optimization.Interfaces Namespace
Upper bound of this variable.

**Namespace:** [Optimization.Interfaces](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>double</code> UpperBound { get; set; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td></td>
</tr>
<tr>
<td>Property</td>
<td>UpperBound As <code>Double</code></td>
</tr>
<tr>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>Set</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
</tr>
<tr>
<td>property</td>
<td><code>double</code> UpperBound {</td>
</tr>
<tr>
<td></td>
<td><code>double</code> get ();</td>
</tr>
<tr>
<td></td>
<td>void set (<code>double</code> value);</td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
</tbody>
</table>
See Also

IVariable Interface
Optimization.Interfaces Namespace
Gets or sets the value for this variable. This field can be used to assign solution values from a solver to the variables in the original model.

**Namespace**: Optimization.Interfaces  
**Assembly**: Optimization.Framework (in Optimization.Framework.dll)  
**Version**: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>double Value { get; set; }</code></td>
</tr>
</tbody>
</table>
| **Visual Basic** | `Property Value As Double
  Get
  Set` |
| **Visual C++** | `property double Value {
  double get ();
  void set (double value);
} ` |

### Field Value

The value.
See Also

IVariable Interface
Optimization.Interfaces Namespace
Interfaces and classes solvers need to implement to work with the framework
<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>🍃 SetupHelpers</td>
<td></td>
</tr>
</tbody>
</table>
## Enumerations

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ModelStatus</td>
<td>Status of a solved model instance.</td>
</tr>
<tr>
<td>SolutionStatus</td>
<td>Status of a solution instance.</td>
</tr>
</tbody>
</table>
Status of a solved model instance.

Namespace: Optimization.Solver
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public enum ModelStatus</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Enumeration ModelStatus</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public enum class ModelStatus</code></td>
</tr>
</tbody>
</table>
### Members

<table>
<thead>
<tr>
<th>Member name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unknown</td>
<td>0</td>
<td>Status of the solved model instance is unknown, e.g. because the solver has too little time to prove the model instance.</td>
</tr>
<tr>
<td>Unbounded</td>
<td>1</td>
<td>The solved model instance is unbounded.</td>
</tr>
<tr>
<td>Infeasible</td>
<td>2</td>
<td>The solved model instance is infeasible.</td>
</tr>
<tr>
<td>Feasible</td>
<td>3</td>
<td>The solved model instance is feasible.</td>
</tr>
<tr>
<td>InfOrUnbd</td>
<td>4</td>
<td>The solved model instance is infeasible or unbounded.</td>
</tr>
</tbody>
</table>
See Also

Optimization.Solver Namespace
Optimization Framework

SetupHelpers Class


Namespace: Optimization.Solver
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public static class SetupHelpers</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public NotInheritable Class SetupHelpers</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public ref class SetupHelpers abstract sealed</code></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
Optimization.Solver.SetupHelpers
See Also

SetupHelpers Members
Optimization.Solver Namespace
The **Setup Helpers** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetPathForSolverLib</td>
<td></td>
</tr>
</tbody>
</table>
See Also

- SetupHelpers Class
- Optimization.Solver Namespace
The **SetupHelpers** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetPathForSolverLib</td>
<td></td>
</tr>
</tbody>
</table>
See Also

SetupHelpers Class
Optimization.Solver Namespace
SetupHelpers.SetPathForSolverLib Method

Namespace: Optimization.Solver
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0


SetupHelpers Class See Also Send Feedback
### Syntax

#### C#

```csharp
public static void SetPathForSolverLib(
    string solvername
)
```

#### Visual Basic

```vbnet
Public Shared Sub SetPathForSolverLib ( _
    solvername As String _
)
```

#### Visual C++

```cpp
public:
static void SetPathForSolverLib(
    String^ solvername
)
```

### Parameters

**solvername**

Type: **System.String**

See Also

SetupHelpers Class
Optimization.Solver Namespace
Status of a solution instance.

**Namespace:** Optimization.Solver  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
<tr>
<td><code>public enum SolutionStatus</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td></td>
</tr>
<tr>
<td><code>Public Enumeration SolutionStatus</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
</tr>
<tr>
<td><code>public enum class SolutionStatus</code></td>
<td></td>
</tr>
</tbody>
</table>
## Members

<table>
<thead>
<tr>
<th>Member name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NoSolutionValues</td>
<td>0</td>
<td>The solution instance contains no variable and objective values, e.g. because the solved model instance is infeasible.</td>
</tr>
<tr>
<td>FeasibleContinuousRelaxation</td>
<td>1</td>
<td>The solution instance contains variable and objective values feasible for the continuous relaxation of the solved model instance.</td>
</tr>
<tr>
<td>OptimalContinuousRelaxation</td>
<td>2</td>
<td>The solution instance contains variable and objective values optimal for the continuous relaxation of the solved model instance.</td>
</tr>
<tr>
<td>Feasible</td>
<td>3</td>
<td>The solution instance contains variable and objective values feasible for the solved model instance.</td>
</tr>
<tr>
<td>ProbablyLocalOptimal</td>
<td>4</td>
<td>The solution instance contains variable and objective values probably local optimal for the solved model instance with the used ISolver.</td>
</tr>
<tr>
<td>LocalOptimal</td>
<td>5</td>
<td>The solution instance contains variable and objective values local optimal for the solved model instance with the used ISolver.</td>
</tr>
<tr>
<td>Optimal</td>
<td>6</td>
<td>The solution instance contains variable and objective values optimal for the solved model instance.</td>
</tr>
</tbody>
</table>
See Also

Optimization.Solver Namespace
Namespace for the Cplex solver wrapper classes
<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CplexSolver</td>
<td>Represents a CPLEX solver instance for mathematical programming problems.</td>
</tr>
<tr>
<td>CplexSolverConfiguration</td>
<td>Configuration for a Cplex solver instance</td>
</tr>
</tbody>
</table>
Optimization Framework

CplexSolver Class

Members  See Also  Send Feedback

Represents a CPLEX solver instance for mathematical programming problems.

**Namespace:** [Optimization.Solver.Cplex](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
<tr>
<td>public class CplexSolver : ICanManipulateANativeSolver, ICanSetCutOff, ISolver, ICanSolve&lt;Model, Solution, IAbortable</td>
<td></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td></td>
</tr>
<tr>
<td>Public Class CplexSolver _ Implements ICanManipulateANativeSolver, ICanSolve&lt;Model, Solution, IAbortable</td>
<td></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
</tr>
<tr>
<td>public ref class CplexSolver : ICanManipulateANativeSolver, ICanSetCutOff, ISolver, ICanSolve&lt;Model, Solution, IAbortable</td>
<td></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
See Also

CplexSolver Members
Optimization.Solver.Cplex Namespace
The **CplexSolver** type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CplexSolver()</td>
<td>Initializes a new instance of the CplexSolver class.</td>
</tr>
<tr>
<td>CplexSolver(SolverConfiguration)</td>
<td>Initializes a new instance of the CplexSolver class.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abort</td>
<td>If this solver instance is busy abort the run as soon as possible, or do nothing if this solver instance is not busy.</td>
</tr>
<tr>
<td>ClearLastModel</td>
<td>Deletes the internal datastructures of this solver instance.</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>Solve</td>
<td>Solves the given model. Optimizes if model contains at least one objective.</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td>The configuration of this solver instance.</td>
</tr>
<tr>
<td>IsBusy</td>
<td>Is this solver instance busy?</td>
</tr>
<tr>
<td>Output</td>
<td>Gets or sets the output textwriter. By setting this property you are able to redirect the solver's output.</td>
</tr>
<tr>
<td>OutputFile</td>
<td>Gets or sets the output file name. Based on the file extension an .lp file or .mps file will be written.</td>
</tr>
</tbody>
</table>
See Also

CplexSolver Class
Optimization.Solver.Cplex Namespace
Optimization Framework

CplexSolver Constructor

CplexSolver Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CplexSolver()</td>
<td>Initializes a new instance of the CplexSolver class.</td>
</tr>
<tr>
<td>CplexSolver(SolverConfiguration)</td>
<td>Initializes a new instance of the CplexSolver class.</td>
</tr>
</tbody>
</table>
See Also

CplexSolver Class
CplexSolver Members
Optimization.Solver.Cplex Namespace
Initializes a new instance of the CplexSolver class.

**Namespace:** Optimization.Solver.Cplex  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>public CplexSolver()</code></td>
<td><code>Public Sub New</code></td>
<td><code>public: CplexSolver()</code></td>
</tr>
</tbody>
</table>
See Also

CplexSolver Class
CplexSolver Overload
Optimization.Solver.Cplex Namespace
Initializes a new instance of the **CplexSolver** class.

**Namespace:** [Optimization.Solver.Cplex](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public CplexSolver(
    SolverConfiguration configuration
)
```

### Visual Basic

```vbnet
Public Sub New (_
    configuration As SolverConfiguration _
)
```

### Visual C++

```cpp
public:
CplexSolver(
    SolverConfiguration^ configuration
)
```

## Parameters

- **configuration**
  - Type: `Optimization::SolverConfiguration`
  - The configuration.
See Also

CplexSolver Class
CplexSolver Overload
Optimization.Solver.Cplex Namespace
The **CplexSolver** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Abort</strong></td>
<td>If this solver instance is busy abort the run as soon as possible, or do nothing if this solver instance is not busy.</td>
</tr>
<tr>
<td><strong>ClearLastModel</strong></td>
<td>Deletes the internal datastructures of this solver instance.</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>Solve</strong></td>
<td>Solves the given <code>model</code>. Optimizes if <code>model</code> contains at least one objective.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>)</td>
</tr>
</tbody>
</table>
See Also

- CplexSolver Class
- Optimization.Solver.Cplex Namespace
If this solver instance is busy abort the run as soon as possible, or do nothing if this solver instance is not busy.

Namespace: Optimization.Solver.Cplex
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>public void Abort()</td>
<td>Public Sub Abort</td>
<td>public: virtual void Abort() sealed</td>
</tr>
</tbody>
</table>

**Implements**

IAborableAbort()
See Also

CplexSolver Class
Optimization.Solver.Cplex Namespace
Optimization Framework

CplexSolver.ClearLastModel Method

CplexSolver Class See Also Send Feedback

Deletes the internal datastructures of this solver instance.

Namespace: Optimization.Solver.Cplex
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
</table>
| public void ClearLastModel() | Public Sub ClearLastModel         | public:
|                              |                                   | virtual void ClearLastModel()     |
|                              |                                   | sealed                            |

**Implements**

ISolver.ClearLastModel()
# Exceptions

<table>
<thead>
<tr>
<th>Exception</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>System.InvalidOperationException</code></td>
<td>If this solver instance is busy.</td>
</tr>
</tbody>
</table>
See Also

CplexSolver Class
Optimization.Solver.Cplex Namespace
Solves the given model. Optimizes if model contains at least one objective.

**Namespace:** [Optimization.Solver.Cplex](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

**C#**

```csharp
public Solution Solve(
    Model model,
    IDictionary<string, double> variableValues
)
```

**Visual Basic**

```vbnet
Public Function Solve ( _
    model As Model, _
    variableValues As IDictionary(Of String, Double) ) As Solution
```

**Visual C++**

```cpp
public:
    virtual Solution^ Solve( 
        Model^ model, 
        IDictionary<String^, double>^ variableValues 
    ) sealed
```

### Parameters

- **model**
  Type: `Optimization.Model`
  The model to solve.

- **variableValues**
  Type: `System.Collections.Generic.IDictionary(String, Double)`
  Initial values for all or a subset of variables in `model`.

### Return Value

**Implements**

ICanSolve(M, S).Solve(M, IDictionary(String, Double))
## Exceptions

<table>
<thead>
<tr>
<th>Exception</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>System.NotSupportedException</td>
<td>If this solver instance not supports solving.</td>
</tr>
<tr>
<td>System.InvalidOperationException</td>
<td>If this solver instance is busy.</td>
</tr>
<tr>
<td>System.ArgumentException</td>
<td>If this solver instance cannot handle the kind of <em>model</em>.</td>
</tr>
</tbody>
</table>
See Also

CplexSolver Class
Optimization.Solver.Cplex Namespace
The **CplexSolver** type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td>The configuration of this solver instance.</td>
</tr>
<tr>
<td>IsBusy</td>
<td>Is this solver instance busy?</td>
</tr>
<tr>
<td>Output</td>
<td>Gets or sets the output textwriter. By setting this property you are able to redirect the solver's output.</td>
</tr>
<tr>
<td>OutputFile</td>
<td>Gets or sets the output file name. Based on the file extension an .lp file or .mps file will be written.</td>
</tr>
</tbody>
</table>
See Also

CplexSolver Class
Optimization.Solver.Cplex Namespace
The configuration of this solver instance.

**Namespace:** [Optimization.Solver.Cplex](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public SolverConfiguration Configuration { get; set; }</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Property Configuration As SolverConfiguration Get Set</code></td>
</tr>
</tbody>
</table>
| **Visual C++**     | `public:
virtual property SolverConfiguration^ Configuration = SolverConfiguration^ get () sealed;
void set (SolverConfiguration^ value) sealed;` |

**Implements**

`ISolver.Configuration`
See Also

CplexSolver Class
Optimization.Solver.Cplex Namespace
Is this solver instance busy?

**Namespace:** [Optimization.Solver.Cplex](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
</table>
| `public  bool  IsBusy  {  get;  internal  set;  }` | `Public  Property  IsBusy  As  Boolean
Get
Friend  Set` | `public:
virtual  property  bool  IsBusy  {
    bool  get ()  sealed;
    internal:  void  set (bool  value)  sealed;
    }` |

**Implements**

`ISolver.IsBusy`
See Also

CplexSolver Class
Optimization.Solver.Cplex Namespace
Gets or sets the output textwriter. By setting this property you are able to redirect the solver's output.

**Namespace:** Optimization.Solver.Cplex  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public TextWriter Output { get; set; }</code></td>
</tr>
</tbody>
</table>
| **Visual Basic** | `Public Property Output As TextWriter
   Get
   Set`             |
| **Visual C++** | `public:
   property TextWriter^ Output {
      TextWriter^ get ();
      void set (TextWriter^ value);` |

### Field Value

The output.
See Also

CplexSolver Class
Optimization.Solver.Cplex Namespace
Gets or sets the output file name. Based on the file extension an .lp file or .mps file will be written.

**Namespace:** Optimization.Solver.Cplex  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

**C#**

```csharp
public FileInfo OutputFile { get; set; }
```

**Visual Basic**

```vbnet
Public Property OutputFile As FileInfo
    Get
        Set
```  

**Visual C++**

```cpp
public:
property FileInfo^ OutputFile {
    FileInfo^ get ();
    void set (FileInfo^ value);
}
```

### Field Value

The output file.
See Also

CplexSolver Class
Optimization.Solver.Cplex Namespace
Configuration for a Cplex solver instance

**Namespace:** Optimization.Solver.Cplex  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public class CplexSolverConfiguration : SolverConfiguration</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Class CplexSolverConfiguration _ Inherits SolverConfiguration</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public ref class CplexSolverConfiguration : public SolverConfiguration</code></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

- System.Object
- Optimization.SolverConfiguration
See Also

CplexSolverConfiguration Members
Optimization.Solver.Cplex Namespace
The `CplexSolverConfiguration` type exposes the following members.
# Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CplexSolverConfiguration()</td>
<td>Initializes a new instance of the CplexSolverConfiguration class.</td>
</tr>
<tr>
<td>CplexSolverConfiguration(String, Boolean)</td>
<td>Initializes a new instance of the CplexSolverConfiguration class.</td>
</tr>
</tbody>
</table>
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is equal to the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.type">Type</a> of the current instance. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="https://docs.microsoft.com/en-us/dotnet/api/system.string">String</a> that represents the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CallbackEndpoint</td>
<td>This endpoint will be used to enable communication between solvers (Inherited from <code>SolverConfiguration</code>).</td>
</tr>
<tr>
<td>GetGlobalIncumbent</td>
<td>(Inherited from <code>SolverConfiguration</code>).</td>
</tr>
<tr>
<td>MIPGap</td>
<td>Relative MIP optimality gap (MIP only). The MIP solver will terminate (with an optimal result) when the relative gap between the lower and upper objective bound is less than MIPGap times the upper bound.</td>
</tr>
<tr>
<td>MIPGapAbs</td>
<td>Sets an absolute tolerance on the gap between the best integer objective and the objective of the best node remaining. When this difference falls below the value of this parameter, the mixed integer optimization is stopped.</td>
</tr>
<tr>
<td>NewBestBoundFound</td>
<td>This action is executed whenever a new best bound is found (Inherited from <code>SolverConfiguration</code>).</td>
</tr>
<tr>
<td>NewIncumbentFound</td>
<td>This action is executed whenever a new incumbent is found (Inherited from <code>SolverConfiguration</code>).</td>
</tr>
<tr>
<td>OptimalityTolerance</td>
<td>Influences the reduced-cost tolerance for optimality. This parameter governs how closely CPLEX must approach the theoretically optimal solution.</td>
</tr>
<tr>
<td>PartialProblemsNeeded</td>
<td>A method that decides if a partial problem should be created. (Inherited from <code>SolverConfiguration</code>).</td>
</tr>
<tr>
<td>SendPartialProblems</td>
<td>A method that sends partial problems over the wire (Inherited from <code>SolverConfiguration</code>).</td>
</tr>
<tr>
<td>TimeLimit</td>
<td>Sets the maximum time, in seconds, for a</td>
</tr>
</tbody>
</table>
call to an optimizer. This time limit applies also to the conflict refiner. The time is measured in terms of either CPU time or elapsed time, according to the setting of the clock type parameter (CPX_PARAM_CLOCKTYPE, ClockType). The time limit for an optimizer applies to the sum of all its steps, such as preprocessing, crossover, and internal calls to other optimizers. In a sequence of calls to optimizers, the limit is not cumulative but applies to each call individually. For example, if you set a time limit of 10 seconds, and you call mipopt twice then there could be a total of (at most) 20 seconds of running time if each call consumes its maximum allotment.

| UseHeuristicCallback | Indicates if the solver should activate its provided HeuristicCallback. This will make the solver use injected solutions from other solvers (Inherited from SolverConfiguration.) |
See Also

CplexSolverConfiguration Class
Optimization.Solver.Cplex Namespace
CplexSolverConfiguration Constructor

CplexSolverConfiguration Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CplexSolverConfiguration()</td>
<td>Initializes a new instance of the CplexSolverConfiguration class.</td>
</tr>
<tr>
<td>CplexSolverConfiguration(String, Boolean)</td>
<td>Initializes a new instance of the CplexSolverConfiguration class.</td>
</tr>
</tbody>
</table>
See Also

CplexSolverConfiguration Class
CplexSolverConfiguration Members
Optimization.Solver.Cplex Namespace
Initializes a new instance of the `CplexSolverConfiguration` class.

**Namespace:** Optimization.Solver.Cplex

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public CplexSolverConfiguration()</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Sub New</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: CplexSolverConfiguration()</td>
</tr>
</tbody>
</table>
See Also

CplexSolverConfiguration Class
CplexSolverConfiguration Overload
Optimization.Solver.Cplex Namespace
Initializes a new instance of the `CplexSolverConfiguration` class.

**Namespace:** [Optimization.Solver.Cplex](https://example.com)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
```csharp
public CplexSolverConfiguration(
    string callbackendpoint,
    bool useHeuristicCallback
)
```

```visual_basic
Public Sub New ( _
    callbackendpoint As String, _
    useHeuristicCallback As Boolean _
)
```

```visual_c++
public:
CplexSolverConfiguration(
    String^ callbackendpoint,
    bool useHeuristicCallback
)
```

**Parameters**

*callbackendpoint*
Type: `System.String`
The callback endpoint.

*useHeuristicCallback*
Type: `System.Boolean`
If set to `true` [use heuristic callback].
See Also

CplexSolverConfiguration Class
CplexSolverConfiguration Overload
Optimization.Solver.Cplex Namespace
The **CplexSolverConfiguration** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <strong>Type</strong> of the current instance. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
</tbody>
</table>
See Also

CplexSolverConfiguration Class
Optimization.Solver.Cplex Namespace
The `CplexSolverConfiguration` type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CallbackEndpoint</strong></td>
<td>This endpoint will be used to enable communication between solvers (Inherited from <a href="#">SolverConfiguration</a>.)</td>
</tr>
<tr>
<td><strong>GetGlobalIncumbent</strong></td>
<td>(Inherited from <a href="#">SolverConfiguration</a>.)</td>
</tr>
<tr>
<td><strong>MIPGap</strong></td>
<td>Relative MIP optimality gap (MIP only). The MIP solver will terminate (with an optimal result) when the relative gap between the lower and upper objective bound is less than MIPGap times the upper bound.</td>
</tr>
<tr>
<td><strong>MIPGapAbs</strong></td>
<td>Sets an absolute tolerance on the gap between the best integer objective and the objective of the best node remaining. When this difference falls below the value of this parameter, the mixed integer optimization is stopped.</td>
</tr>
<tr>
<td><strong>NewBestBoundFound</strong></td>
<td>This action is executed whenever a new best bound is found (Inherited from <a href="#">SolverConfiguration</a>.)</td>
</tr>
<tr>
<td><strong>NewIncumbentFound</strong></td>
<td>This action is executed whenever a new incumbent is found (Inherited from <a href="#">SolverConfiguration</a>.)</td>
</tr>
<tr>
<td><strong>OptimalityTolerance</strong></td>
<td>Influences the reduced-cost tolerance for optimality. This parameter governs how closely CPLEX must approach the theoretically optimal solution.</td>
</tr>
<tr>
<td><strong>PartialProblemsNeeded</strong></td>
<td>A method that decides if a partial problem should be created. (Inherited from <a href="#">SolverConfiguration</a>.)</td>
</tr>
<tr>
<td><strong>SendPartialProblems</strong></td>
<td>A method that sends partial problems over the wire (Inherited from <a href="#">SolverConfiguration</a>.)</td>
</tr>
<tr>
<td><strong>TimeLimit</strong></td>
<td>Sets the maximum time, in seconds, for a</td>
</tr>
</tbody>
</table>
call to an optimizer. This time limit applies also to the conflict refiner. The time is measured in terms of either CPU time or elapsed time, according to the setting of the clock type parameter (CPX_PARAM_CLOCKTYPE, ClockType). The time limit for an optimizer applies to the sum of all its steps, such as preprocessing, crossover, and internal calls to other optimizers. In a sequence of calls to optimizers, the limit is not cumulative but applies to each call individually. For example, if you set a time limit of 10 seconds, and you call mipopt twice then there could be a total of (at most) 20 seconds of running time if each call consumes its maximum allotment.

| UseHeuristicCallback | Indicates if the solver should activate its provided HeuristicCallback. This will make the solver use injected solutions from other solvers (Inherited from SolverConfiguration.) |
See Also

CplexSolverConfiguration Class
Optimization.Solver.Cplex Namespace
Relative MIP optimality gap (MIP only). The MIP solver will terminate (with an optimal result) when the relative gap between the lower and upper objective bound is less than MIPGap times the upper bound.

**Namespace:** Optimization.Solver.Cplex

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public double MIPGap { get; set; }
```

Visual Basic

```vbnet
Public Property MIPGap As Double
    Get
        Get
    End Get
    Set
        Set (value) = MIPGap
    End Set
End Property
```

Visual C++

```cpp
public:
    property double MIPGap
    {
        double get ();
        void set (double value);
    }
```

Field Value

Sets a relative tolerance on the gap between the best integer objective and the objective of the best node remaining. When the value \(|\text{bestnode-bestinteger}|/(1e-10+|\text{bestinteger}|)\) falls below the value of this parameter, the mixed integer optimization is stopped. For example, to instruct CPLEX to stop as soon as it has found a feasible integer solution proved to be within five percent of optimal, set the relative mipgap tolerance to 0.05.
See Also

CplexSolverConfiguration Class
Optimization.Solver.Cplex Namespace
CplexSolverConfiguration.MIPGapAbs Property

Sets an absolute tolerance on the gap between the best integer objective and the objective of the best node remaining. When this difference falls below the value of this parameter, the mixed integer optimization is stopped.

Namespace: Optimization.Solver.Cplex
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th></th>
<th>C#</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>public double MIPGapAbs { get; set; }</td>
</tr>
<tr>
<td>Visual Basic</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Public Property MIPGapAbs As Double Get</td>
</tr>
<tr>
<td></td>
<td>Set</td>
</tr>
<tr>
<td>Visual C++</td>
<td></td>
</tr>
<tr>
<td></td>
<td>public: double MIPGapAbs {</td>
</tr>
<tr>
<td></td>
<td>double get ();</td>
</tr>
<tr>
<td></td>
<td>void set (double value);</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

### Field Value

The MIP gap abs.
See Also

CplexSolverConfiguration Class
Optimization.Solver.Cplex Namespace
Influences the reduced-cost tolerance for optimality. This parameter governs how closely CPLEX must approach the theoretically optimal solution.

Namespace: Optimization.Solver.Cplex
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public double OptimalityTolerance { get; set; }
```

**Visual Basic**

```vbnet
Public Property OptimalityTolerance As Double
    Get
    Set
```

**Visual C++**

```cpp
public:
    property double OptimalityTolerance {
        double get ();
        void set (double value);
    }
```

### Field Value

The optimality tolerance.
See Also

CplexSolverConfiguration Class
Optimization.Solver.Cplex Namespace
CplexSolverConfiguration.TimeLimit Property

Sets the maximum time, in seconds, for a call to an optimizer. This time limit applies also to the conflict refiner. The time is measured in terms of either CPU time or elapsed time, according to the setting of the clock type parameter (CPX_PARAM_CLOCKTYPE, ClockType). The time limit for an optimizer applies to the sum of all its steps, such as preprocessing, crossover, and internal calls to other optimizers. In a sequence of calls to optimizers, the limit is not cumulative but applies to each call individually. For example, if you set a time limit of 10 seconds, and you call mipopt twice then there could be a total of (at most) 20 seconds of running time if each call consumes its maximum allotment.

Namespace: Optimization.Solver.Cplex
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public double TimeLimit { get; set; }
```

Visual Basic

```vbnet
Public Property TimeLimit As Double
    Get
        Set
```

Visual C++

```c++
public:
    property double TimeLimit {
        double get ();
        void set (double value);
    }
```

Field Value

The time limit.
See Also

CplexSolverConfiguration Class
Optimization.Solver.Cplex Namespace
Namespace for different types of events solvers can fire and handle
## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>🚀 NewIncumbentFound</td>
<td>This event is fired whenever a new incumbent was found during the solution process</td>
</tr>
</tbody>
</table>
This event is fired whenever a new incumbent was found during the solution process.

**Namespace:** Optimization.Solver.Events  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

**C#**

```csharp
public class NewIncumbentFound : ISolverEvent
```

**Visual Basic**

```vbnet
Public Class NewIncumbentFound Implements ISolverEvent
```

**Visual C++**

```cpp
public ref class NewIncumbentFound : ISolverEvent
```
Inheritance Hierarchy

System.Object

Optimization.Solver.Events.NewIncumbentFound
See Also

NewIncumbentFound Members
Optimization.Solver.Events Namespace
The *NewIncumbentFound* type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewIncumbentFound</td>
<td>Initializes a new instance of the NewIncumbentFound class</td>
</tr>
</tbody>
</table>
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IncumbentCoefficients</td>
<td>Gets or sets the incumbent coefficients.</td>
</tr>
<tr>
<td>ObjectiveValue</td>
<td>Gets or sets the objective value.</td>
</tr>
<tr>
<td>Solver</td>
<td>Gets or sets the solver.</td>
</tr>
</tbody>
</table>
See Also

NewIncumbentFound Class
Optimization.Solver.Events Namespace
NewIncumbentFound Constructor

Initializes a new instance of the **NewIncumbentFound** class

**Namespace:** Optimization.Solver.Events  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
</tr>
<tr>
<td>public NewIncumbentFound()</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
</tr>
<tr>
<td>Public Sub New</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
</tr>
<tr>
<td>public: NewIncumbentFound()</td>
</tr>
</tbody>
</table>
See Also

**NewIncumbentFound Class**

**Optimization.Solver.Events Namespace**
The **NewIncumbentFound** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

NewIncumbentFound Class
Optimization.Solver.Events Namespace
The **NewIncumbentFound** type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IncumbentCoefficients</td>
<td>Gets or sets the incumbent coefficients.</td>
</tr>
<tr>
<td>ObjectiveValue</td>
<td>Gets or sets the objective value.</td>
</tr>
<tr>
<td>Solver</td>
<td>Gets or sets the solver.</td>
</tr>
</tbody>
</table>
See Also

NewIncumbentFound Class
Optimization.Solver.Events Namespace
Optimization Framework

NewIncumbentFound.IncumbentCoefficients Property

Gets or sets the incumbent coefficients.

**Namespace:** [Optimization.Solver.Event](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public double[] IncumbentCoefficients { get; set; }
```

Visual Basic

```vbnet
Public Property IncumbentCoefficients As Double()
    Get
    Set
```

Visual C++

```cpp
public:
property array<double>^ IncumbentCoefficients {
    array<double>^ get ();
    void set (array<double>^ value);
}
```

Field Value

The incumbent coefficients.
See Also

NewIncumbentFound Class
Optimization.Solver.Events Namespace
Optimization Framework

NewIncumbentFound.ObjectiveValue Property

Gets or sets the objective value.

**Namespace:** [Optimization.Solver.Events](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>public double ObjectiveValue { get; set; }</td>
<td>ObjectiveValue accessor in C#</td>
</tr>
<tr>
<td>Visual Basic</td>
<td>Public Property ObjectiveValue As Double Get Set</td>
<td>ObjectiveValue accessor in Visual Basic</td>
</tr>
<tr>
<td>Visual C++</td>
<td>public: property double ObjectiveValue { double get (); void set (double value); }</td>
<td>ObjectiveValue accessor in Visual C++</td>
</tr>
</tbody>
</table>

### Field Value

The objective value.
See Also

NewIncumbentFound Class
Optimization.Solver.Events Namespace
Optimization Framework

NewIncumbentFound.Solver Property

Gets or sets the solver.

**Namespace:** Optimization.Solver.Events

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

public string Solver { get; set; }

Visual Basic

Public Property Solver As String
    Get
    Set

Visual C++

public:
    property String^ Solver {
        String^ get ();
        void set (String^ value);
    }

FieldValue

The solver.
See Also

NewIncumbentFound Class  
Optimization.Solver.Events Namespace
Namespace for the Gurobi solver wrapper classes
<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>🍇 GurobiSolverConfiguration</td>
<td>The configuration for Gurobi</td>
</tr>
</tbody>
</table>
# Enumerations

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SolverStatus</td>
<td>The Status of the Gurobi solver</td>
</tr>
</tbody>
</table>
The configuration for Gurobi

**Namespace:** [Optimization.Solver.Gurobi](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public class GurobiSolverConfiguration : SolverConfiguration</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Class GurobiSolverConfiguration _ Inherits SolverConfiguration</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public ref class GurobiSolverConfiguration : public</code></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

- `System.Object`
- `Optimization.SolverConfiguration`
See Also

GurobiSolverConfiguration Members
Optimization.Solver.Gurobi Namespace
The **GurobiSolverConfiguration** type exposes the following members.
### Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>GurobiSolverConfiguration()</code></td>
<td>Initializes a new instance of the <code>GurobiSolverConfiguration</code> class.</td>
</tr>
<tr>
<td><code>GurobiSolverConfiguration(String, Boolean, String)</code></td>
<td>Initializes a new instance of the <code>GurobiSolverConfiguration</code> class with the specified values.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CalculateDuals</strong></td>
<td>Gets or sets a value indicating whether duals shall be calculated or not.</td>
</tr>
<tr>
<td><strong>CallbackEndpoint</strong></td>
<td>This endpoint will be used to enable communication between solvers (Inherited from <a href="#">SolverConfiguration</a>).</td>
</tr>
<tr>
<td><strong>ComputeIIS</strong></td>
<td>Gets or sets a value indicating whether the IIS shall be computed. If false, the solution will return null in the ConflictingSet property - default is true.</td>
</tr>
<tr>
<td><strong>ConfigFile</strong></td>
<td>Gets or sets the config file.</td>
</tr>
<tr>
<td><strong>FeasibilityTolerance</strong></td>
<td>Primal feasibility tolerance. All constraints must be satisfied to a tolerance of FeasibilityTol.</td>
</tr>
<tr>
<td><strong>GetGlobalIncumbent</strong></td>
<td>(Inherited from <a href="#">SolverConfiguration</a>).</td>
</tr>
<tr>
<td><strong>IntegerFeasibilityTolerance</strong></td>
<td>Integer feasibility tolerance (MIP only). An integrality restriction on a variable is considered satisfied when the variable's value is less than IntFeasTol from the nearest integer value.</td>
</tr>
<tr>
<td><strong>MIPGap</strong></td>
<td>Relative MIP optimality gap (MIP only). The MIP solver will terminate (with an optimal result) when the relative gap between the lower and upper objective bound is less than MIPGap times the upper bound.</td>
</tr>
<tr>
<td><strong>MIPGapAbs</strong></td>
<td>Absolute MIP optimality gap (MIP only). The MIP solver will terminate (with an optimal result) when the absolute gap between the lower and upper objective bound is less than MIPGapAbs.</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>NewBestBoundFound</strong></td>
<td>This action is executed whenever a new best bound is found. (Inherited from <em>SolverConfiguration</em>.)</td>
</tr>
<tr>
<td><strong>NewIncumbentFound</strong></td>
<td>This action is executed whenever a new incumbent is found. (Inherited from <em>SolverConfiguration</em>.)</td>
</tr>
<tr>
<td><strong>OptimalityTolerance</strong></td>
<td>Dual feasibility tolerance. Reduced costs must all be smaller than OptimalityTol in the improving direction in order for a model to be declared optimal.</td>
</tr>
<tr>
<td><strong>OutputFile</strong></td>
<td>Gets or sets the output file</td>
</tr>
<tr>
<td><strong>PartialProblemsNeeded</strong></td>
<td>A method that decides if a partial problem should be created. (Inherited from <em>SolverConfiguration</em>.)</td>
</tr>
<tr>
<td><strong>SendPartialProblems</strong></td>
<td>A method that sends partial problems over the wire. (Inherited from <em>SolverConfiguration</em>.)</td>
</tr>
<tr>
<td><strong>TimeLimit</strong></td>
<td>Limits the total time expended (in seconds). Note that all runtimes in the Gurobi Optimizer are wall-clock times.</td>
</tr>
<tr>
<td><strong>UseHeuristicCallback</strong></td>
<td>Indicates if the solver should activate its provided HeuristicCallback. This will make the solver use injected solutions from other solvers. (Inherited from <em>SolverConfiguration</em>.)</td>
</tr>
</tbody>
</table>
See Also

GurobiSolverConfiguration Class
Optimization.Solver.Gurobi Namespace
Optimization Framework

GurobiSolverConfiguration Constructor

GurobiSolverConfiguration Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GurobiSolverConfiguration()</td>
<td>Initializes a new instance of the GurobiSolverConfiguration class.</td>
</tr>
<tr>
<td>GurobiSolverConfiguration(String, Boolean, String)</td>
<td>Initializes a new instance of the GurobiSolverConfiguration class with the specified values.</td>
</tr>
</tbody>
</table>
See Also

GurobiSolverConfiguration Class
GurobiSolverConfiguration Members
Optimization.Solver.Gurobi Namespace
Optimization Framework

GurobiSolverConfiguration Constructor

GurobiSolverConfiguration Class See Also Send Feedback

Initializes a new instance of the GurobiSolverConfiguration class.

Namespace: Optimization.Solver.Gurobi

Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public GurobiSolverConfiguration()
```

**Visual Basic**

```vbnet
Public Sub New
```

**Visual C++**

```cpp
public:
GurobiSolverConfiguration()
```
See Also

GurobiSolverConfiguration Class
GurobiSolverConfiguration Overload
Optimization.Solver.Gurobi Namespace
Initializes a new instance of the `GurobiSolverConfiguration` class with the specified values.

**Namespace:** Optimization.Solver.Gurobi  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#**            | `public GurobiSolverConfiguration(  
|                   |     string callbackendpoint,  
|                   |     bool useHeuristicCallback,  
|                   |     string configFile  
|                   | )`                                                                    |
| **Visual Basic**  | `Public Sub New ( _  
|                   |     callbackendpoint As String, _  
|                   |     useHeuristicCallback As Boolean, _  
|                   |     configFile As String _  
|                   | )`                                                                    |
| **Visual C++**    | `public:  
|                   | GurobiSolverConfiguration(  
|                   |     String^ callbackendpoint,  
|                   |     bool useHeuristicCallback,  
|                   |     String^ configFile  
|                   | )`                                                                    |

### Parameters

- **callbackendpoint**
  - Type: `System.String`
  - The callbackendpoint.

- **useHeuristicCallback**
  - Type: `System.Boolean`
  - if set to `true` [use heuristic callback].
configFile
Type: System.String
The config file.
See Also

GurobiSolverConfiguration Class
GurobiSolverConfiguration Overload
Optimization.Solver.Gurobi Namespace
The **GurobiSolverConfiguration** type exposes the following members.
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
</tbody>
</table>
See Also

GurobiSolverConfiguration Class
Optimization.Solver.Gurobi Namespace
The `GurobiSolverConfiguration` type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalculateDuals</td>
<td>Gets or sets a value indicating whether duals shall be calculated or not.</td>
</tr>
<tr>
<td>CallbackEndpoint</td>
<td>This endpoint will be used to enable communication between solvers (Inherited from <a href="#">SolverConfiguration</a>.)</td>
</tr>
<tr>
<td>ComputeIIS</td>
<td>Gets or sets a value indicating whether the IIS shall be computed. If false, the solution will return null in the ConflictingSet property - default is true</td>
</tr>
<tr>
<td>ConfigFile</td>
<td>Gets or sets the config file.</td>
</tr>
<tr>
<td>FeasibilityTolerance</td>
<td>Primal feasibility tolerance. All constraints must be satisfied to a tolerance of FeasibilityTol.</td>
</tr>
<tr>
<td>GetGlobalIncumbent</td>
<td>(Inherited from <a href="#">SolverConfiguration</a>.)</td>
</tr>
<tr>
<td>IntegerFeasibilityTolerance</td>
<td>Integer feasibility tolerance (MIP only). An integrality restriction on a variable is considered satisfied when the variable's value is less than IntFeasTol from the nearest integer value.</td>
</tr>
<tr>
<td>MIPGap</td>
<td>Relative MIP optimality gap (MIP only). The MIP solver will terminate (with an optimal result) when the relative gap between the lower and upper objective bound is less than MIPGap times the upper bound.</td>
</tr>
<tr>
<td>MIPGapAbs</td>
<td>Absolute MIP optimality gap (MIP only). The MIP solver will terminate (with an optimal result) when the absolute gap between the lower and upper objective bound is less than MIPGapAbs.</td>
</tr>
<tr>
<td><strong>NewBestBoundFound</strong></td>
<td>This action is executed whenever a new best bound is found (Inherited from <a href="#">SolverConfiguration</a>).</td>
</tr>
<tr>
<td><strong>NewIncumbentFound</strong></td>
<td>This action is executed whenever a new incumbent is found (Inherited from <a href="#">SolverConfiguration</a>).</td>
</tr>
<tr>
<td><strong>OptimalityTolerance</strong></td>
<td>Dual feasibility tolerance. Reduced costs must all be smaller than OptimalityTol in the improving direction in order for a model to be declared optimal.</td>
</tr>
<tr>
<td><strong>OutputFile</strong></td>
<td>Gets or sets the output file</td>
</tr>
<tr>
<td><strong>PartialProblemsNeeded</strong></td>
<td>A method that decides if a partial problem should be created. (Inherited from <a href="#">SolverConfiguration</a>).</td>
</tr>
<tr>
<td><strong>SendPartialProblems</strong></td>
<td>A method that sends partial problems over the wire (Inherited from <a href="#">SolverConfiguration</a>).</td>
</tr>
<tr>
<td><strong>TimeLimit</strong></td>
<td>Limits the total time expended (in seconds). Note that all runtimes in the Gurobi Optimizer are wall-clock times.</td>
</tr>
<tr>
<td><strong>UseHeuristicCallback</strong></td>
<td>Indicates if the solver should activate its provided HeuristicCallback. This will make the solver use injected solutions from other solvers (Inherited from <a href="#">SolverConfiguration</a>).</td>
</tr>
</tbody>
</table>
See Also

GurobiSolverConfiguration Class
Optimization.Solver.Gurobi Namespace
Gets or sets a value indicating whether duals shall be calculated or not.

**Namespace:** Optimization.Solver.Gurobi

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public bool CalculateDuals { get; set; }
```

**Visual Basic**

```vbnet
Public Property CalculateDuals As Boolean
    Get
    Set
```

**Visual C++**

```cpp
public:
    property bool CalculateDuals {
        bool get ();
        void set (bool value);
    }
```

### Field Value

*true* if duals should be calculated; otherwise, *false*. 
See Also

GurobiSolverConfiguration Class
Optimization.Solver.Gurobi Namespace
Gets or sets a value indicating whether the IIS shall be computed. If false, the solution will return null in the ConflictingSet property - default is true

**Namespace:** Optimization.Solver.Gurobi  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th><strong>C#</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>public bool ComputeIIS { get; set; }</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Visual Basic</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Property ComputeIIS As Boolean</td>
</tr>
<tr>
<td>Get</td>
</tr>
<tr>
<td>Set</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Visual C++</strong></th>
</tr>
</thead>
</table>
| public:
property bool ComputeIIS {
  bool get ();
  void set (bool value);
} |

### Field Value

true if IIS shall be computed; otherwise, false.
See Also

GurobiSolverConfiguration Class
Optimization.Solver.Gurobi Namespace
Gets or sets the config file.

**Namespace:** Optimization.Solver.Gurobi  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#**     | ```
public string ConfigFile { get; set; }
``` |
| **Visual Basic** | ```
Public Property ConfigFile As String
    Get
    Set
``` |
| **Visual C++** | ```
public:
    property String^ ConfigFile { 
        String^ get ();
        void set (String^ value);
    }
``` |

### Field Value

The config file.
See Also

GurobiSolverConfiguration Class
Optimization.Solver.Gurobi Namespace
Primal feasibility tolerance. All constraints must be satisfied to a tolerance of FeasibilityTol.

**Namespace:** Optimization.Solver.Gurobi  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>public double FeasibilityTolerance { get; set; }</code></td>
<td><code>Public Property FeasibilityTolerance As Double Get Set</code></td>
<td><code>public: property double FeasibilityTolerance {</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>double get ();</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>void set (double value);</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>}</code></td>
</tr>
</tbody>
</table>

### Field Value

The feasibility tolerance.
See Also

GurobiSolverConfiguration Class
Optimization.Solver.Gurobi Namespace
Integer feasibility tolerance (MIP only). An integrality restriction on a variable is considered satisfied when the variable's value is less than IntFeasTol from the nearest integer value.

**Namespace:** Optimization.Solver.Gurobi

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public double IntegerFeasibilityTolerance { get; set; }</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Property IntegerFeasibilityTolerance As Double</code>&lt;br&gt;<code>Get</code>&lt;br&gt;<code>Set</code></td>
</tr>
</tbody>
</table>
| **Visual C++** | `public:
    property double IntegerFeasibilityTolerance { double get ();
        void set (double value);` |

## Field Value

The integer feasibility tolerance.
See Also

GurobiSolverConfiguration Class
Optimization.Solver.Gurobi Namespace
GurobiSolverConfiguration.MIPGap Property

Relative MIP optimality gap (MIP only). The MIP solver will terminate (with an optimal result) when the relative gap between the lower and upper objective bound is less than MIPGap times the upper bound.

Namespace: Optimization.Solver.Gurobi
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>public double MIPGap { get; set; }</td>
</tr>
<tr>
<td>Visual Basic</td>
<td>Public Property MIPGap As Double Get Set</td>
</tr>
<tr>
<td>Visual C++</td>
<td>public: property double MIPGap { double get (); void set (double value); }</td>
</tr>
</tbody>
</table>

### Field Value

The MIP gap.
See Also

GurobiSolverConfiguration Class
Optimization.Solver.Gurobi Namespace
Absolute MIP optimality gap (MIP only). The MIP solver will terminate (with an optimal result) when the absolute gap between the lower and upper objective bound is less than MIPGapAbs.

**Namespace:** [Optimization.Solver.Gurobi](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public double MIPGapAbs { get; set; }
```

**Visual Basic**

```vbnet
Public Property MIPGapAbs As Double
    Get
    Set
```

**Visual C++**

```cpp
public:
    property double MIPGapAbs {
        double get ();
        void set (double value);
    }
```

### Field Value

The MIP gap abs.
See Also

GurobiSolverConfiguration Class
Optimization.Solver.Gurobi Namespace
Dual feasibility tolerance. Reduced costs must all be smaller than OptimalityTol in the improving direction in order for a model to be declared optimal.

Namespace: Optimization.Solver.Gurobi
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public double OptimalityTolerance { get; set; }
```

**Visual Basic**

```vbnet
Public Property OptimalityTolerance As Double
    Get
    Set
```

**Visual C++**

```cpp
public:
    property double OptimalityTolerance {
        double get ()
        void set (double value);
    }
```

### Field Value

The optimality tolerance.
See Also

GurobiSolverConfiguration Class
Optimization.Solver.Gurobi Namespace
Gets or sets the output file

**Namespace:** [Optimization.Solver.Gurobi](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
## Syntax

### C#

```csharp
public FileInfo OutputFile { get; set; }
```

### Visual Basic

```vbnet
Public Property OutputFile As FileInfo
    Get
    Set
```

### Visual C++

```cpp
public:
property FileInfo^ OutputFile { 
    FileInfo^ get ();
    void set (FileInfo^ value);
}
```

## Field Value

The file extension decides what type of file is written. The only valid extensions are lp and mps.
See Also

GurobiSolverConfiguration Class
Optimization.Solver.Gurobi Namespace
Limits the total time expended (in seconds). Note that all runtimes in the Gurobi Optimizer are wall-clock times.

**Namespace:** Optimization.Solver.Gurobi  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public double TimeLimit { get; set; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Property TimeLimit As Double</td>
</tr>
<tr>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>Set</td>
</tr>
</tbody>
</table>
| **Visual C++** | public:
|            | property double TimeLimit {
|            |     double get ();
|            |     void set (double value);
|            | } |

### Field Value

The time limit.
See Also

GurobiSolverConfiguration Class
Optimization.Solver.Gurobi Namespace
The Status of the Gurobi solver

**Namespace:** [Optimization.Solver.Gurobi](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>public enum SolverStatus</td>
<td>Public Enumeration SolverStatus</td>
<td>public enum class SolverStatus</td>
</tr>
</tbody>
</table>
## Members

<table>
<thead>
<tr>
<th>Member name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cutoff</td>
<td>6</td>
<td>Cutoff</td>
</tr>
<tr>
<td>Infeasible</td>
<td>3</td>
<td>Infeasible</td>
</tr>
<tr>
<td>InfOrUnbd</td>
<td>4</td>
<td>Infinity or Unbounded</td>
</tr>
<tr>
<td>Interrupted</td>
<td>11</td>
<td>Interrupted</td>
</tr>
<tr>
<td>IterationLimit</td>
<td>7</td>
<td>IterationLimit</td>
</tr>
<tr>
<td>Loaded</td>
<td>1</td>
<td>Loaded</td>
</tr>
<tr>
<td>NodeLimit</td>
<td>8</td>
<td>NodeLimit</td>
</tr>
<tr>
<td>Numeric</td>
<td>12</td>
<td>Numeric</td>
</tr>
<tr>
<td>Optimal</td>
<td>2</td>
<td>Optimal</td>
</tr>
<tr>
<td>SolutionLimit</td>
<td>10</td>
<td>SolutionLimit</td>
</tr>
<tr>
<td>TimeLimit</td>
<td>9</td>
<td>TimeLimit</td>
</tr>
<tr>
<td>Unbounded</td>
<td>5</td>
<td>Unbounded</td>
</tr>
<tr>
<td>Suboptimal</td>
<td>13</td>
<td>Suboptimal</td>
</tr>
</tbody>
</table>
See Also

Optimization.Solver.Gurobi Namespace
Optimization Framework

Optimization.Solver.Gurobi40 Namespace

### Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>🌐 GurobiSolver</td>
<td></td>
</tr>
</tbody>
</table>

Namespace: Optimization.Solver.Gurobi40
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public class GurobiSolver : ISolver, ICanSolve&lt;Model, Solution&gt;, IAbortable, ICanManipulateANativeSolver</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Class GurobiSolver Implements ISolver, ICanSolve(Of Model, Solution), IAbortable, ICanHandleSolverEventOfType(NewIncumbentFound)</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public ref class GurobiSolver : ISolver, ICanSolve&lt;Model^, Solution^&gt;, IAbortable, ICanManipulateANativeSolver</td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
See Also

GurobiSolver Members
Optimization.Solver.Gurobi40 Namespace
The `GurobiSolver` type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GurobiSolver</td>
<td>Initializes a new instance of the GurobiSolver class</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abort</td>
<td></td>
</tr>
<tr>
<td>ClearLastModel</td>
<td></td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td>HandleEvent</td>
<td></td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>Solve</td>
<td></td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a String that represents the current Object. (Inherited from Object.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td></td>
</tr>
<tr>
<td>IsBusy</td>
<td></td>
</tr>
</tbody>
</table>
See Also

GurobiSolver Class
Optimization.Solver.Gurobi40 Namespace
Initializes a new instance of the `GurobiSolver` class

**Namespace:** `Optimization.Solver.Gurobi40`  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
</table>
| **C#**   | ```
public GurobiSolver(
    GurobiSolverConfiguration configuration
)
``` |
| **Visual Basic** | ```
Public Sub New ( _
    configuration As GurobiSolverConfiguration _
)
``` |
| **Visual C++**  | ```
public:
GurobiSolver(
    GurobiSolverConfiguration^ configuration
)
``` |

### Parameters

- **configuration**
See Also

GurobiSolver Class
Optimization.Solver.Gurobi40 Namespace
The **GurobiSolver** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abort</td>
<td></td>
</tr>
<tr>
<td>ClearLastModel</td>
<td></td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is equal to the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.type">Type</a> of the current instance. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td>HandleEvent</td>
<td></td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td>Solve</td>
<td></td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <a href="https://docs.microsoft.com/en-us/dotnet/api/system.string">String</a> that represents the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
</tbody>
</table>


See Also

GurobiSolver Class
Optimization.Solver.Gurobi40 Namespace
[Missing <summary> documentation for

Namespace: Optimization.Solver.Gurobi40
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

**C#**

```csharp
public void Abort()
```

**Visual Basic**

```vbnet
Public Sub Abort
```

**Visual C++**

```cpp
public:
virtual void Abort() sealed
```

**Implements**

```csharp
IAbortable.Abort()
```
See Also

GurobiSolver Class
Optimization.Solver.Gurobi40 Namespace
Optimization Framework

GurobiSolver.ClearLastModel Method

GurobiSolver Class See Also Send Feedback


Namespace: Optimization.Solver.Gurobi40
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public void ClearLastModel()</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Sub ClearLastModel</code></td>
</tr>
</tbody>
</table>
| **Visual C++** | `public:
virtual void ClearLastModel() sealed`                          |

**Implements**

`ISolver.ClearLastModel()`
See Also

GurobiSolver Class
Optimization.Solver.Gurobi40 Namespace
**Optimization Framework**

**GurobiSolver.HandleEvent Method**

*GurobiSolver Class  See Also  Send Feedback*

**[Missing <summary> documentation for**


**Namespace:** Optimization.Solver.Gurobi40

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#**   | ```
public void HandleEvent(
    NewIncumbentFound solverEvent
 )
``` |
| **Visual Basic** | ```
Public Sub HandleEvent (_
    solverEvent As NewIncumbentFound _
 )
``` |
| **Visual C++** | ```
public:
    virtual void HandleEvent(
        NewIncumbentFound^ solverEvent
    ) sealed
``` |

### Parameters

- **solverEvent**
  - Type: `Optimization.Solver.Events.NewIncumbentFound`

### Implements

- `ICanHandleSolverEventOfType<T>,HandleEvent(T)`
See Also

GurobiSolver Class
Optimization.Solver.Gurobi40 Namespace
### Syntax

#### C#

```csharp
public Solution Solve(
    Model model,
    IDictionary<string, double> variableValues
)
```

#### Visual Basic

```vbnet
Public Function Solve (_
    model As Model, _
    variableValues As IDictionary(Of String, Double)
) As Solution
```

#### Visual C++

```cpp
public:
    virtual Solution^ Solve(
        Model^ model,
        IDictionary<String^, double>^ variableValues
    ) sealed
```

### Parameters

- **model**
  Type: `Optimization.Model`

- **variableValues**
  Type: `System.Collections.Generic.IDictionary<String, Double>`
Return Value


Implements

ICanSolve(M, S).Solve(M, IDictionary(String, Double))
See Also

GurobiSolver Class
Optimization.Solver.Gurobi40 Namespace
The GurobiSolver type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td></td>
</tr>
<tr>
<td>IsBusy</td>
<td></td>
</tr>
</tbody>
</table>
See Also

GurobiSolver Class
Optimization.Solver.Gurobi40 Namespace
Optimization Framework

GurobiSolver.Configuration Property

GurobiSolver Class  See Also  Send Feedback

[Missing <summary> documentation for

Namespace: Optimization.Solver.Gurobi40
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public SolverConfiguration Configuration { get; set; }
```

### Visual Basic

```vbnet
Public Property Configuration As SolverConfiguration
    Get
    Set
```

### Visual C++

```cpp
public:
    virtual property SolverConfiguration^ Configuration :
    SolverConfiguration^ get () sealed;
    void set (SolverConfiguration^ value) sealed;
}
```

## Implements

`ISolver.Configuration`
See Also

GurobiSolver Class
Optimization.Solver.Gurobi40 Namespace
GurobiSolver.IsBusy Property


Namespace: Optimization.Solver.Gurobi40
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

**C#**

```csharp
public bool IsBusy { get; private set; }
```

**Visual Basic**

```vbnet
Public Property IsBusy As Boolean
    Get
        Private Set
    End Property
```

**Visual C++**

```cpp
public:
    virtual property bool IsBusy {
        bool get () sealed;
        private: void set (bool value) sealed;
    }
```

**Implements**

`ISolver.IsBusy`
See Also

GurobiSolver Class
Optimization.Solver.Gurobi40 Namespace
### Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GurobiSolver</td>
<td></td>
</tr>
</tbody>
</table>

Namespace: Optimization.Solver.Gurobi45
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>public class GurobiSolver : ISolver, ICanSolve&lt;Model, Solution&gt;, IAbortable, ICanHandleSolverEventOfType, ICanManipulateANativeSolver</td>
<td>Public Class GurobiSolver _ Implements ISolver, ICanSolve(Of Model, Solution), IAbortable, ICanHandleSolverEventOfType(Of NewIncumbentFound), ICanManipulateANativeSolver</td>
<td>public ref class GurobiSolver : ISolver, ICanSolve&lt;Model^, Solution^&gt;, IAbortable, ICanHandleSolverEventOfType, ICanManipulateANativeSolver</td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object

See Also

GurobiSolver Members
Optimization.Solver.Gurobi45 Namespace
The `GurobiSolver` type exposes the following members.
### Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GurobiSolver</td>
<td>Initializes a new instance of the GurobiSolver class</td>
</tr>
</tbody>
</table>
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abort</td>
<td></td>
</tr>
<tr>
<td>ClearLastModel</td>
<td></td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is equal to the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.type">Type</a> of the current instance. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td>HandleEvent</td>
<td></td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td>Solve</td>
<td></td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <a href="https://docs.microsoft.com/en-us/dotnet/api/system.string">String</a> that represents the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td></td>
</tr>
<tr>
<td>IsBusy</td>
<td></td>
</tr>
</tbody>
</table>
See Also

GurobiSolver Class
Optimization.Solver.Gurobi45 Namespace
GurobiSolver Constructor

Initializes a new instance of the GurobiSolver class

Namespace: Optimization.Solver.Gurobi45
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public GurobiSolver(
  GurobiSolverConfiguration configuration
)
```

### Visual Basic

```vbnet
Public Sub New (_
  configuration As GurobiSolverConfiguration _
)
```

### Visual C++

```cpp
public:
GurobiSolver(
  GurobiSolverConfiguration^ configuration
)
```

## Parameters

- **configuration**
  - [Missing <param name="configuration"/> documentation for "M:Optimization.Solver.Gurobi45.GurobiSolver.#ctor(Optimization.Solve...
See Also

GurobiSolver Class
Optimization.Solver.Gurobi45 Namespace
The **GurobiSolver** type exposes the following members.
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abort</td>
<td></td>
</tr>
<tr>
<td>ClearLastModel</td>
<td></td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup</td>
</tr>
<tr>
<td></td>
<td>operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>HandleEvent</td>
<td></td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>Solve</td>
<td></td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a String that represents the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
</tbody>
</table>
See Also

GurobiSolver Class
Optimization.Solver.Gurobi45 Namespace
GurobiSolver.Abort Method

Namespace: Optimization.Solver.Gurobi45
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public void Abort()
```

### Visual Basic

```vbnet
Public Sub Abort
```

### Visual C++

```cpp
public:
    virtual void Abort() sealed
```

## Implements

IAborable::Abort()
See Also

GurobiSolver Class
Optimization.Solver.Gurobi45 Namespace
Optimization Framework

GurobiSolver.ClearLastModel Method

GurobiSolver Class See Also Send Feedback


Namespace: Optimization.Solver.Gurobi45
Assembly: Optimization/Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public void ClearLastModel()</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Sub ClearLastModel</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: virtual void ClearLastModel() sealed</td>
</tr>
</tbody>
</table>

**Implements**

[ISolver.ClearLastModel()](#)
See Also

GurobiSolver Class
Optimization.Solver.Gurobi45 Namespace
GurobiSolver.HandleEvent Method

GurobiSolver Class  See Also  Send Feedback

[Missing <summary> documentation for

Namespace: Optimization.Solver.Gurobi45
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version:
0.3.1036.0
## Syntax

### C#

```csharp
public void HandleEvent(  
    NewIncumbentFound solverEvent  
)
```

### Visual Basic

```vbnet
Public Sub HandleEvent ( _
        solverEvent As NewIncumbentFound _
    )
```

### Visual C++

```cpp
public:
    virtual void HandleEvent(  
        NewIncumbentFound^ solverEvent  
    ) sealed
```

## Parameters

**solverEvent**

Type: `Optimization.Solver.Events.NewIncumbentFound`


## Implements

`ICanHandleSolverEventOfType(T), HandleEvent(T)"`
See Also

GurobiSolver Class
Optimization.Solver.Gurobi45 Namespace
GurobiSolver.Solve Method

Namespace: Optimization.Solver.Gurobi45
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>public Solution Solve(</td>
<td>Public Function Solve ( _</td>
<td>public: virtual Solution^ Solve(</td>
</tr>
<tr>
<td>Model model,</td>
<td>model As Model, _</td>
<td>Model^ model,</td>
</tr>
<tr>
<td>IDictionary&lt;string, double&gt; variableValues</td>
<td>variableValues As IDictionary(Of String, Double) As Solution)</td>
<td></td>
</tr>
<tr>
<td>)</td>
<td>)</td>
<td>) sealed</td>
</tr>
</tbody>
</table>

### Parameters

- **model**
  - Type: `Optimization.Model`
  IDictionary<string, double>)"]

- **variableValues**
  - Type: `System.Collections.Generic.IDictionary<String, Double>`
  IDictionary<string, double>)"]
Return Value

[Missing <returns> documentation for

Implements

ICanSolve(M, S).Solve(M, IDictionary(String, Double))
See Also

GurobiSolver Class
Optimization.Solver.Gurobi45 Namespace
The **GurobiSolver** type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td></td>
</tr>
<tr>
<td>IsBusy</td>
<td></td>
</tr>
</tbody>
</table>
See Also

GurobiSolver Class
Optimization.Solver.Gurobi45 Namespace
Optimization Framework

GurobiSolver.Configuration Property

GurobiSolver Class See Also Send Feedback

[Missing <summary> documentation for

Namespace: Optimization.Solver.Gurobi45
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public SolverConfiguration Configuration { get; set; }
```

**Visual Basic**

```vbnet
Public Property Configuration As SolverConfiguration
    Get
        Get
    Set
        Set
```

**Visual C++**

```cpp
public:
    virtual property SolverConfiguration^ Configuration :
        SolverConfiguration^ get () sealed;
        void set (SolverConfiguration^ value) sealed;
}
```

### Implements

**ISolver.Configuration**
See Also

GurobiSolver Class
Optimization.Solver.Gurobi45 Namespace
Optimization Framework

GurobiSolver.IsBusy Property


Namespace: Optimization.Solver.Gurobi45
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public <strong>bool</strong> IsBusy { get; private set; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Property <strong>IsBusy</strong> As <strong>Boolean</strong></td>
</tr>
<tr>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>Private Set</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: virtual property <strong>bool</strong> IsBusy {</td>
</tr>
<tr>
<td></td>
<td><strong>bool</strong> get () sealed;</td>
</tr>
<tr>
<td></td>
<td>private: void set (<strong>bool</strong> value) sealed;</td>
</tr>
</tbody>
</table>

**Implements**

**ISolver.IsBusy**
See Also

GurobiSolver Class
Optimization.Solver.Gurobi45 Namespace
Optimization Framework

Optimization.Solver.Gurobi46 Namespace

Send Feedback

### Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>🌟 GurobiSolver</td>
<td></td>
</tr>
</tbody>
</table>
GurobiSolver Class


Namespace: Optimization.Solver.Gurobi46
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
</tr>
<tr>
<td>public class GurobiSolver : ISolver, ICanSolve&lt;Model, Solution&gt;, IAbortable, ICanManipulateANativeSolver</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
</tr>
</tbody>
</table>
| Public Class GurobiSolver _
  Implements ISolver, ICanSolve(Of Model, Solution), IAbortable, ICanHandleSolverEventOfType(Of NewIncumbentFound) |
| **Visual C++** |
| public ref class GurobiSolver : ISolver, ICanSolve<Model^, Solution^>, IAbortable, ICanHandleANativeSolver |
Inheritance Hierarchy

System.Object
See Also

GurobiSolver Members
Optimization.Solver.Gurobi46 Namespace
The **GurobiSolver** type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GurobiSolver</td>
<td>Initializes a new instance of the GurobiSolver class</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Abort</strong></td>
<td></td>
</tr>
<tr>
<td><strong>ClearLastModel</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <strong>Type</strong> of the current instance. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>HandleEvent</strong></td>
<td></td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>Solve</strong></td>
<td></td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td></td>
</tr>
<tr>
<td>IsBusy</td>
<td></td>
</tr>
</tbody>
</table>
See Also

GurobiSolver Class
Optimization.Solver.Gurobi46 Namespace
Initializes a new instance of the `GurobiSolver` class

**Namespace:** `Optimization.Solver.Gurobi46`  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

**C#**

```csharp
public GurobiSolver(
    GurobiSolverConfiguration configuration
)
```

**Visual Basic**

```vbnet
Public Sub New (_
    configuration As GurobiSolverConfiguration _
)
```

**Visual C++**

```csharp
public:
GurobiSolver(
    GurobiSolverConfiguration^ configuration
)
```

### Parameters

- **configuration**

See Also

GurobiSolver Class
Optimization.Solver.Gurobi46 Namespace
The **GurobiSolver** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Abort</strong></td>
<td></td>
</tr>
<tr>
<td><strong>ClearLastModel</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a> (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>HandleEvent</strong></td>
<td></td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a> (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td><strong>Solve</strong></td>
<td></td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a> (Inherited from <a href="#">Object</a>)</td>
</tr>
</tbody>
</table>
See Also

GurobiSolver Class
Optimization.Solver.Gurobi46 Namespace
Optimization Framework

GurobiSolverAbort Method

GurobiSolver Class See Also Send Feedback


Namespace: Optimization.Solver.Gurobi46
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th></th>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><code>public void Abort()</code></td>
<td><code>Public Sub Abort</code></td>
<td><code>public: virtual void Abort() sealed</code></td>
</tr>
</tbody>
</table>

**Implements**

`IAborable.Abort()`
See Also

GurobiSolver Class
Optimization.Solver.Gurobi46 Namespace
GurobiSolver.ClearLastModel Method

GurobiSolver Class See Also Send Feedback


Namespace: Optimization.Solver.Gurobi46
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
<tr>
<td><code>public void ClearLastModel()</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td></td>
</tr>
<tr>
<td><code>Public Sub ClearLastModel</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
</tr>
</tbody>
</table>
| `public:
  virtual void ClearLastModel() sealed` |   |

**Implements**

[ISolver.ClearLastModel()](#)
See Also

GurobiSolver Class
Optimization.Solver.Gurobi46 Namespace
Optimization Framework

GurobiSolver.HandleEvent Method

GurobiSolver Class See Also Send Feedback


Namespace: Optimization.Solver.Gurobi46
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| C#         | ```
public void HandleEvent(
    NewIncumbentFound solverEvent
)
``` |
| Visual Basic | ```
Public Sub HandleEvent ( _
    solverEvent As NewIncumbentFound _
)
``` |
| Visual C++ | ```
public:
    virtual void HandleEvent(
        NewIncumbentFound^ solverEvent
    ) sealed
``` |

### Parameters

`solverEvent`  
Type: `Optimization.Solver.Events.NewIncumbentFound`  

### Implements

`ICanHandleSolverEventOfType<T>, HandleEvent<T>`
See Also

GurobiSolver Class
Optimization.Solver.Gurobi46 Namespace
GurobiSolver.Solve Method

Namespace: Optimization.Solver.Gurobi46
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public Solution Solve(
    Model model,
    IDictionary<string, double> variableValues
)
```

#### Visual Basic

```vbnet
Public Function Solve (  
    model As Model,  
    variableValues As IDictionary(Of String, Double)  
) As Solution
```

#### Visual C++

```cpp
public: 
virtual Solution^ Solve(
    Model^ model, 
    IDictionary<ViewSDK::String^, double>^ variableValues 
) sealed
```

### Parameters

- **model**
  Type: `Optimization.Model`

[Missing `<param name="model"/>` documentation for 

- **variableValues**
  Type: `System.Collections.Generic.IDictionary<String, Double>`

[Missing `<param name="variableValues"/>` documentation for 
Return Value


Implements

ICanSolve(M, S).Solve(M, IDictionary(String, Double))
See Also

GurobiSolver Class
Optimization.Solver.Gurobi46 Namespace
The **GurobiSolver** type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td></td>
</tr>
<tr>
<td>IsBusy</td>
<td></td>
</tr>
</tbody>
</table>
See Also

GurobiSolver Class
Optimization.Solver.Gurobi46 Namespace
[Missing <summary> documentation for

Namespace: Optimization.Solver.Gurobi46
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#
```csharp
public SolverConfiguration Configuration { get; set; }
```

#### Visual Basic
```vbnet
Public Property Configuration As SolverConfiguration
    Get
        Get
    End Get
    Set
        Set
    End Set
End Property
```

#### Visual C++
```cpp
public:
    virtual property SolverConfiguration^ Configuration {
        SolverConfiguration^ get () sealed;
        void set (SolverConfiguration^ value) sealed;
    }
```

**Implements**

ISolver.Configuration
See Also

GurobiSolver Class
Optimization.Solver.Gurobi46 Namespace
Optimization Framework

GurobiSolver.IsBusy Property

GurobiSolver Class See Also Send Feedback


Namespace: Optimization.Solver.Gurobi46
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public bool IsBusy { get; private set; }
```

#### Visual Basic

```vbnet
Public Property IsBusy As Boolean
    Get
        Private Set
    End Property
```

#### Visual C++

```cpp
public:
    virtual property bool IsBusy {
        bool get () sealed;
        private: void set (bool value) sealed;
    }
```

### Implements

-ISolver.IsBusy-
See Also

GurobiSolver Class
Optimization.Solver.Gurobi46 Namespace
[Missing <summary> documentation for "N:Optimization.Solver.Interfaces"]
### Interfaces

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IAbortable</td>
<td>If a solver can be aborted it should implement this interface</td>
</tr>
<tr>
<td>ICanHandleSolverEventOfType(T)</td>
<td>Indicates that the implementing class can handle a certain SolverEvent</td>
</tr>
<tr>
<td>ICanSolve(M, S)</td>
<td>Indicates that this solver can solve models of type M and return solutions of type S</td>
</tr>
<tr>
<td>ISolverEvent</td>
<td>An implementing class represents an event that can be fired by a solver</td>
</tr>
</tbody>
</table>
If a solver can be aborted it should implement this interface

**Namespace:** [Optimization.Solver.Interfaces](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
<tr>
<td><code>public interface IAbortable</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td></td>
</tr>
<tr>
<td><code>Public Interface IAbortable</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
</tr>
<tr>
<td><code>public interface class IAbortable</code></td>
<td></td>
</tr>
</tbody>
</table>
See Also

IAabortable Members
Optimization.Solver.Interfaces Namespace
The **IAbortable** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abort</td>
<td>If this solver instance is busy abort the run as soon as possible, or do nothing if this solver instance is not busy.</td>
</tr>
</tbody>
</table>
See Also

IAabortable Interface
Optimization.Solver.Interfaces Namespace
The **IAbortable** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abort</td>
<td>If this solver instance is busy abort the run as soon as possible, or do nothing if this solver instance is not busy.</td>
</tr>
</tbody>
</table>
See Also

IAbortable Interface
Optimization.Solver.Interfaces Namespace
If this solver instance is busy abort the run as soon as possible, or do nothing if this solver instance is not busy.

**Namespace:** Optimization.Solver.Interfaces

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
void Abort()
```

**Visual Basic**

```vbnet
Sub Abort
```

**Visual C++**

```cpp
void Abort()
```
## Exceptions

<table>
<thead>
<tr>
<th>Exception</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>System.NotSupportedException</code></td>
<td>If this solver instance not supports aborting.</td>
</tr>
</tbody>
</table>
See Also

IAbortable Interface
Optimization.Solver.Interfaces Namespace
ICanHandleSolverEventOfType(T) Interface

Indicates that the implementing class can handle a certain SolverEvent

Namespace: Optimization.Solver.Interfaces
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public interface ICanHandleSolverEventOfType<T>
where T : ISolverEvent
```

**Visual Basic**

```vbnet
Public Interface ICanHandleSolverEventOfType(Of T As
```

**Visual C++**

```cpp
generic<typename T>
where T : ISolverEvent
public interface class ICanHandleSolverEventOfType
```

```cpp
```
Type Parameters

\( T \)

[Missing <typeparam name="T"/> documentation for "T:Optimization.Solver.Interfaces.ICanHandleSolverEventOfType`1"]
See Also

ICanHandleSolverEventOfType(T) Members
Optimization.Solver.Interfaces Namespace
The `ICanHandleSolverEventOfType<T>` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HandleEvent</td>
<td>Handles the event. This can for example be an event that is fired whenever a NewIncombent was found during the solution process.</td>
</tr>
</tbody>
</table>
See Also

ICanHandleSolverEventOfType(T) Interface
Optimization.Solver.Interfaces Namespace
The **ICanHandleSolverEventOfType(T)** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HandleEvent</td>
<td>Handles the event. This can for example be an event that is fired whenever a NewIncombent was found during the solution process.</td>
</tr>
</tbody>
</table>
See Also

ICanHandleSolverEventOfType(T) Interface
Optimization.Solver.Interfaces Namespace
ICanHandleSolverEventOfType\(T\).HandleEvent Method

ICanHandleSolverEventOfType\(T\) Interface  See Also  Send Feedback

Handles the event. This can for example be an event that is fired whenever a NewIncomitant was found during the solution process.

**Namespace:** Optimization.Solver.Interfaces  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th></th>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>void HandleEvent(</td>
<td>Sub HandleEvent ( _</td>
<td>void HandleEvent(</td>
</tr>
<tr>
<td></td>
<td>T solverEvent</td>
<td>solverEvent As T _</td>
<td>T solverEvent</td>
</tr>
<tr>
<td></td>
<td>)</td>
<td>)</td>
<td>)</td>
</tr>
</tbody>
</table>

### Parameters

**solverEvent**

Type: T

The solver event.
See Also

ICanHandleSolverEventOfType(T) Interface
Optimization.Solver.Interfaces Namespace
ICanSolve(M, S) Interface

Indicates that this solver can solve models of type M and return solutions of type S

Namespace: Optimization.Solver.Interfaces
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
# Syntax

**C#**

```csharp
public interface ICanSolve<M, S>
```

**Visual Basic**

```vbnet
Public Interface ICanSolve(Of M, S)
```

**Visual C++**

```cpp
generic<typename M, typename S>
public interface class ICanSolve
```
Type Parameters

$M$
The model type

$S$
The solution type
See Also

ICanSolve(M, S) Members
Optimization.Solver.Interfaces Namespace
The `ICanSolve(M, S)` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solve</td>
<td>Solves the given model. Optimizes if model contains at least one objective.</td>
</tr>
</tbody>
</table>
See Also

ICanSolve(M, S) Interface
Optimization.Solver.Interfaces Namespace
Optimization Framework

ICanSolve(M, S) Methods

ICanSolve(M, S) Interface See Also Send Feedback

The ICanSolve(M, S) type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solve</td>
<td>Solves the given <em>model</em>. Optimizes if <em>model</em> contains at least one objective.</td>
</tr>
</tbody>
</table>
See Also

ICanSolve(M, S) Interface
Optimization.Solver.Interfaces Namespace
Solves the given *model*. Optimizes if *model* contains at least one objective.

**Namespace:** [Optimization.Solver.Interfaces](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td><code>S Solve(M model, IDictionary&lt;string, double&gt; variableValues)</code></td>
</tr>
<tr>
<td>Visual Basic</td>
<td><code>Function Solve (_ model As M, _ variableValues As IDictionary(Of String, Double) As S</code></td>
</tr>
<tr>
<td>Visual C++</td>
<td><code>S Solve(M model, IDictionary&lt;String, double&gt; variableValues)</code></td>
</tr>
</tbody>
</table>

### Parameters

- **model**  
  Type: `M`  
  The model to solve.

- **variableValues**  
  Type: `System.Collections.Generic.IDictionary(String, Double)`  
  Initial values for all or a subset of variables in `model`.

### Return Value

[Missing <returns> documentation for]
## Exceptions

<table>
<thead>
<tr>
<th>Exception</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>System.NotSupportedException</code></td>
<td>If this solver instance not supports solving.</td>
</tr>
<tr>
<td><code>System.InvalidOperationException</code></td>
<td>If this solver instance is busy.</td>
</tr>
<tr>
<td><code>System.ArgumentException</code></td>
<td>If this solver instance cannot handle the kind of <em>model</em>.</td>
</tr>
</tbody>
</table>
See Also

ICanSolve(M, S) Interface
Optimization.Solver.Interfaces Namespace
An implementing class represents an event that can be fired by a solver

**Namespace:** Optimization.Solver.Interfaces  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th><strong>C#</strong></th>
<th><strong>Visual Basic</strong></th>
<th><strong>Visual C++</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>public interface ISolverEvent</code></td>
<td><code>Public Interface ISolverEvent</code></td>
<td><code>public interface class ISolverEvent</code></td>
</tr>
</tbody>
</table>
See Also

Optimization.Solver.Interfaces Namespace
[Missing <summary> documentation for "N:Optimization.Solver.MOPS"]
### Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOPSDLLFunctions</td>
<td></td>
</tr>
<tr>
<td>MOPSSolver</td>
<td></td>
</tr>
<tr>
<td>MOPSSolverConfiguration</td>
<td></td>
</tr>
</tbody>
</table>
## Enumerations

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOPSLicenseType</td>
<td></td>
</tr>
</tbody>
</table>
[Missing <summary> documentation for "T:Optimization.Solver.MOPS.MOPSDLLFunctions"]

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public static class MOPSDLLFunctions</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public NotInheritable Class MOPSDLLFunctions</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public ref class MOPSDLLFunctions abstract sealed</code></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

- System.Object
- Optimization.Solver.MOPS.MOPSDLFunctions
See Also

MOPSDLFunctions Members
Optimization.Solver.MOPS Namespace
The **MOPSDLFunctions** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finish</td>
<td></td>
</tr>
<tr>
<td>GenFileNames</td>
<td></td>
</tr>
<tr>
<td>GetIPSolution</td>
<td></td>
</tr>
<tr>
<td>GetLPSolution</td>
<td></td>
</tr>
<tr>
<td>GetMOPSVersion</td>
<td></td>
</tr>
<tr>
<td>GetParameter</td>
<td></td>
</tr>
<tr>
<td>Initialize</td>
<td></td>
</tr>
<tr>
<td>Optimize</td>
<td></td>
</tr>
<tr>
<td>PutModel</td>
<td></td>
</tr>
<tr>
<td>ReadProfile</td>
<td></td>
</tr>
<tr>
<td>SetCallback</td>
<td></td>
</tr>
<tr>
<td>SetLicense</td>
<td></td>
</tr>
<tr>
<td>SetParameter</td>
<td></td>
</tr>
</tbody>
</table>
See Also

MOPSDLLFunctions Class
Optimization.Solver.MOPS Namespace
The **MOPSDLFunctions** type exposes the following members.
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finish</td>
<td></td>
</tr>
<tr>
<td>GenFileNames</td>
<td></td>
</tr>
<tr>
<td>GetIPSolution</td>
<td></td>
</tr>
<tr>
<td>GetLPSolution</td>
<td></td>
</tr>
<tr>
<td>GetMOPSVersion</td>
<td></td>
</tr>
<tr>
<td>GetParameter</td>
<td></td>
</tr>
<tr>
<td>Initialize</td>
<td></td>
</tr>
<tr>
<td>Optimize</td>
<td></td>
</tr>
<tr>
<td>PutModel</td>
<td></td>
</tr>
<tr>
<td>ReadProfile</td>
<td></td>
</tr>
<tr>
<td>SetCallback</td>
<td></td>
</tr>
<tr>
<td>SetLicense</td>
<td></td>
</tr>
<tr>
<td>SetParameter</td>
<td></td>
</tr>
</tbody>
</table>
See Also

MOPSDLLFunctions Class
Optimization.Solver.MOPS Namespace
MOPSDLFunctions.Finish Method

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public static int Finish()</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Shared Function Finish As Integer</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: static int Finish()</td>
</tr>
</tbody>
</table>

### Return Value

See Also

MOPSDLLFunctions Class
Optimization.Solver.MOPS Namespace

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

**C#**

```csharp
public static int GenFileNames(
    string sFileName
)
```

**Visual Basic**

```vbnet
Public Shared Function GenFileNames ( _
    sFileName As String _
) As Integer
```

**Visual C++**

```cpp
public:
static int GenFileNames(
    String^ sFileName
)
```

**Parameters**

`sFileName`
Type: `System.String`


**Return Value**

See Also

MOPSDLFunctions Class
Optimization.Solver.MOPS Namespace
[Missing <summary> documentation for

**Namespace:** Optimization.Solver.MOPS

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public static int GetIPSolution(
    out int iIpSta,
    out double dIpFunct,
    IntPtr daXs,
    IntPtr daDj,
    IntPtr iaSta
)
```

#### Visual Basic

```vbnet
Public Shared Function GetIPSolution ( _
    <OutAttribute> ByRef iIpSta As Integer, _
    <OutAttribute> ByRef dIpFunct As Double, _
    daXs As IntPtr, _
    daDj As IntPtr, _
    iaSta As IntPtr _
) As Integer
```

#### Visual C++

```cpp
public:
static int GetIPSolution(
    int% iIpSta,
    double% dIpFunct,
    IntPtr daXs,
    IntPtr daDj,
    IntPtr iaSta
)
```

### Parameters
iIpSta
Type: System.Int32


dIpFunct
Type: System.Double


daXs
Type: System.IntPtr


daDj
Type: System.IntPtr


iaSta
Type: System.IntPtr


Return Value

See Also

MOPSDLLFunctions Class
Optimization.Solver.MOPS Namespace
Optimization Framework

MOPSDLLFunctions.GetLPSolution Method

MOPSDLLFunctions Class See Also Send Feedback


Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public static int GetLPSolution(
    out int iLpSta,
    out double dLpFunct,
    IntPtr daXs,
    IntPtr daDj,
    IntPtr iaSta
)
```

Visual Basic

```vbnet
Public Shared Function GetLPSolution ( _
    <OutAttribute> ByRef iLpSta As Integer, _
    <OutAttribute> ByRef dLpFunct As Double, _
    daXs As IntPtr, _
    daDj As IntPtr, _
    iaSta As IntPtr _
) As Integer
```

Visual C++

```cpp
public:
static int GetLPSolution(
    [OutAttribute] int% iLpSta,
    [OutAttribute] double% dLpFunct,
    IntPtr daXs,
    IntPtr daDj,
    IntPtr iaSta
)
```

Parameters
**iLpSta**
Type: [System.Int32](https://docs.microsoft.com/en-us/dotnet/api/system.int32)


**dLpFunct**
Type: [System.Double](https://docs.microsoft.com/en-us/dotnet/api/system.double)


**daXs**
Type: [System.IntPtr](https://docs.microsoft.com/en-us/dotnet/api/systemIntPtr)


**daDj**
Type: [System.IntPtr](https://docs.microsoft.com/en-us/dotnet/api/systemIntPtr)


**iaSta**
Type: [System.IntPtr](https://docs.microsoft.com/en-us/dotnet/api/systemIntPtr)


**Return Value**


See Also

- MOPSDLLEFunctions Class
- Optimization.Solver.MOPS Namespace
MOPSDLLFunctions.GetMOPSVersion Method

MOPSDLLFunctions Class


Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

**C#**

```csharp
public static int GetMOPSVersion(
    StringBuilder sbValue
)
```

**Visual Basic**

```vbnet
Public Shared Function GetMOPSVersion ( _
    sbValue As StringBuilder _
) As Integer
```

**Visual C++**

```cpp
public:
static int GetMOPSVersion(
    StringBuilder^ sbValue
)
```

**Parameters**

*sbValue*
Type: `System.Text.StringBuilder`

[Missing `<param name="sbValue"/>` documentation for "M:Optimization.Solver.MOPS.MOPSDLLFunctions.GetMOPSVersion(S"

**Return Value**

[Missing `<returns>` documentation for "M:Optimization.Solver.MOPS.MOPSDLLFunctions.GetMOPSVersion(S"
See Also

MOPSDL.LFunctions Class
Optimization.Solver.MOPS Namespace
MOPSDLLFunctions.GetParameter Method

MOPSDLLFunctions Class  See Also  Send Feedback


Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public static int GetParameter(
    string sParameter,
    StringBuilder sbValue
)
```

### Visual Basic

```vbnet
Public Shared Function GetParameter ( _
    sParameter As String, _
    sbValue As StringBuilder _
) As Integer
```

### Visual C++

```cpp
public:
static int GetParameter(
    String^ sParameter, 
    StringBuilder^ sbValue
)
```

## Parameters

**sParameter**

Type: `System.String`


**sbValue**

Type: `System.Text.StringBuilder`

Return Value


See Also

MOPSDLFunctions Class
Optimization.Solver.MOPS Namespace
MOPSDLFunctions.Initialize Method

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public static int Initialize()</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Shared Function Initialize As Integer</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public: static int Initialize()</code></td>
</tr>
</tbody>
</table>

### Return Value

See Also

MOPSDLLFunctions Class
Optimization.Solver.MOPS Namespace
Optimization Framework

MOPSDLFunctions.Optimize Method

MOPSDLFunctions Class See Also Send Feedback

[Missing <summary> documentation for

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
</tr>
</thead>
<tbody>
<tr>
<td>public static int Optimize(</td>
</tr>
<tr>
<td>int iDirection,</td>
</tr>
<tr>
<td>out int iStatus,</td>
</tr>
<tr>
<td>out int iPhase,</td>
</tr>
<tr>
<td>out double dObjFunc</td>
</tr>
<tr>
<td>)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual Basic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Shared Function Optimize (</td>
</tr>
<tr>
<td>iDirection As Integer,</td>
</tr>
<tr>
<td>&lt;OutAttribute&gt; ByRef iStatus As Integer,</td>
</tr>
<tr>
<td>&lt;OutAttribute&gt; ByRef iPhase As Integer,</td>
</tr>
<tr>
<td>&lt;OutAttribute&gt; ByRef dObjFunc As Double</td>
</tr>
<tr>
<td>) As Integer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>public:</td>
</tr>
<tr>
<td>static int Optimize(</td>
</tr>
<tr>
<td>int iDirection,</td>
</tr>
<tr>
<td>[OutAttribute] int% iStatus,</td>
</tr>
<tr>
<td>[OutAttribute] int% iPhase,</td>
</tr>
<tr>
<td>[OutAttribute] double% dObjFunc</td>
</tr>
<tr>
<td>)</td>
</tr>
</tbody>
</table>

### Parameters

**iDirection**
Type: System.Int32

[Missing <param name="iDirection"/> documentation for]

**iStatus**
Type: **System.Int32**

**iPhase**
Type: **System.Int32**

**dObjFunc**
Type: **System.Double**

**Return Value**

See Also

MOPSDLLFunctions Class
Optimization.Solver.MOPS Namespace
MOPSDLFunctions.PutModel Method
MOPSDLFunctions Class See Also Send Feedback

[Missing <summary> documentation for

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public static int PutModel(
    int iIntype,
    double dINF,
    int iNRows,
    int iNCols,
    int iNNz,
    IntPtr iaRowIndices,
    IntPtr iaColIndices,
    IntPtr daNZs,
    IntPtr daLBs,
    IntPtr daUBs,
    IntPtr daCosts,
    IntPtr iaColTypes
)
```

#### Visual Basic

```vbnet
Public Shared Function PutModel ( _
    iIntype As Integer, _
    dINF As Double, _
    iNRows As Integer, _
    iNCols As Integer, _
    iNNz As Integer, _
    iaRowIndices As IntPtr, _
    iaColIndices As IntPtr, _
    daNZs As IntPtr, _
    daLBs As IntPtr, _
    daUBs As IntPtr, _
    daCosts As IntPtr, _
    iaColTypes As IntPtr _
) As Integer
```

#### Visual C++

```c++
public static int PutModel(
    int iIntype,
    double dINF,
    int iNRows,
    int iNCols,
    int iNNz,
    IntPtr iaRowIndices,
    IntPtr iaColIndices,
    IntPtr daNZs,
    IntPtr daLBs,
    IntPtr daUBs,
    IntPtr daCosts,
    IntPtr iaColTypes
)
```
public:
static int PutModel(
    int iIntype,
    double dINF,
    int iNRows,
    int iNCols,
    int iNNz,
    IntPtr iaRowIndices,
    IntPtr iaColIndices,
    IntPtr daNZs,
    IntPtr daLBs,
    IntPtr daUBs,
    IntPtr daCosts,
    IntPtr iaColTypes
)

Parameters

iIntype
Type: System.Int32

dINF
Type: System.Double

iNRows
Type: System.Int32

iNCols
Type: System.Int32
iNNz
Type: System.Int32
[Missing <param name="iNNz"/> documentation for "M:Optimization.Solver.MOPS.MOPSDLLFunctions.PutModel(System.Int32,
 iaRowIndices
Type: System.IntPtr
[Missing <param name="iaRowIndices"/> documentation for "M:Optimization.Solver.MOPS.MOPSDLLFunctions.PutModel(System.Int32,
 iaColIndices
Type: System.IntPtr
[Missing <param name="iaColIndices"/> documentation for "M:Optimization.Solver.MOPS.MOPSDLLFunctions.PutModel(System.Int32,
 daNZs
Type: System.IntPtr
[Missing <param name="daNZs"/> documentation for "M:Optimization.Solver.MOPS.MOPSDLLFunctions.PutModel(System.Int32,
 daLBs
Type: System.IntPtr
[Missing <param name="daLBs"/> documentation for "M:Optimization.Solver.MOPS.MOPSDLLFunctions.PutModel(System.Int32,
 daUBs
Type: System.IntPtr
[Missing <param name="daUBs"/> documentation for "M:Optimization.Solver.MOPS.MOPSDLLFunctions.PutModel(System.Int32,
 daCosts
Type: System.IntPtr
 iaColTypes
Type: System.IntPtr
[Missing <param name="iaColTypes"/> documentation for "M:Optimization.Solver.MOPS.MOPSDLLFunctions.PutModel(System.Int32,

Return Value
See Also

- MOPSDLFunctions Class
- Optimization.Solver.MOPS Namespace
Optimization Framework

MOPSDLFunctions.ReadProfile Method

MOPSDLFunctions Class See Also Send Feedback

[Missing <summary> documentation for
"M:Optimization.Solver.MOPS.MOPSDLFunctions.ReadProfile(System.String)"

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Parameters

sFileName
Type: System.String

[Missing <param name="sFileName"/> documentation for
"M:Optimization.Solver.MOPS.MOPSDLLFunctions.ReadProfile(System.

Return Value

[Missing <returns> documentation for
"M:Optimization.Solver.MOPS.MOPSDLLFunctions.ReadProfile(System.

Syntax

C#

public static int ReadProfile(
    string sFileName
)

Visual Basic

Public Shared Function ReadProfile ( _
    sFileName As String _
) As Integer

Visual C++

public:
static int ReadProfile(
    String^ sFileName
)
See Also

MOPSDLLFunctions Class
Optimization.Solver.MOPS Namespace
Optimization Framework

MOPSDLFFunctions.SetCallback Method


Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public static int SetCallback(
    Delegate pCB,
    int iCBType
)
```

Visual Basic

```vbnet
Public Shared Function SetCallback ( _
    pCB As Delegate, _
    iCBType As Integer _
) As Integer
```

Visual C++

```c++
public:
static int SetCallback(
    Delegate^ pCB,
    int iCBType
)
```

Parameters

**pCB**
Type: `System.Delegate`

**iCBType**
Type: `System.Int32`
Return Value

[Missing <returns> documentation for "M:Optimization.Solver.MOPS.MOPSDLLFunctions.SetCallback(System..."
See Also

MOPSDLLFunctions Class
Optimization.Solver.MOPS Namespace
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
</tr>
</thead>
<tbody>
<tr>
<td>public static int SetLicense(</td>
</tr>
<tr>
<td>int iLicenseNUmber</td>
</tr>
<tr>
<td>)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual Basic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Shared Function SetLicense ( _</td>
</tr>
<tr>
<td>iLicenseNUmber As Integer _</td>
</tr>
<tr>
<td>) As Integer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>public:</td>
</tr>
<tr>
<td>static int SetLicense(</td>
</tr>
<tr>
<td>int iLicenseNUmber</td>
</tr>
<tr>
<td>)</td>
</tr>
</tbody>
</table>

### Parameters

- **iLicenseNUmber**
  - Type: System.Int32
  - [Missing <param name="iLicenseNUmber"/> documentation for "M:Optimization.Solver.MOPS.MOPSDLLFunctions.SetLicense(System.I...]

### Return Value

- [Missing <returns> documentation for "M:Optimization.Solver.MOPS.MOPSDLLFunctions.SetLicense(System.I...]
See Also

MOPSDLFunctions Class
Optimization.Solver.MOPS Namespace
MOPSDLLFunctions.SetParameter Method

MOPSDLLFunctions Class  See Also  Send Feedback


**Namespace:** Optimization.Solver.MOPS

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

**C#**

```csharp
public static int SetParameter(string sParameter)
```

**Visual Basic**

```vbnet
Public Shared Function SetParameter (_
    sParameter As String _
) As Integer
```

**Visual C++**

```cpp
public:
static int SetParameter( _
    String^ sParameter
)
```

## Parameters

**sParameter**

Type: `System.String`

[Missing `<param name="sParameter"/>` documentation for "M:Optimization.Solver.MOPS.MOPSDLLFunctions.SetParameter(System.String)"

## Return Value


See Also

MOPSDLLFunctions Class
Optimization.Solver.MOPS Namespace
[Missing <summary> documentation for "T:Optimization.Solver.MOPS.MOPSLicenseType"]

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public enum MOPSLicenseType</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Enumeration MOPSLicenseType</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public enum class MOPSLicenseType</code></td>
</tr>
</tbody>
</table>
### Members

<table>
<thead>
<tr>
<th>Member name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ValidNoLimits</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>InvalidDongleMissing</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>InvalidTimeLimit</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>ValidTimeLimit</td>
<td>-3</td>
<td></td>
</tr>
<tr>
<td>ValidSizeLimit</td>
<td>-4</td>
<td></td>
</tr>
<tr>
<td>ValidTimeAndSizeLimit</td>
<td>-5</td>
<td></td>
</tr>
<tr>
<td>ValidNoSolution</td>
<td>-6</td>
<td></td>
</tr>
</tbody>
</table>
See Also

Optimization.Solver.MOPS Namespace
[Missing <summary> documentation for "T:Optimization.Solver.MOPS.MOPSSolver"]

**Namespace:** Optimization.Solver.MOPS

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
</tr>
</thead>
<tbody>
<tr>
<td>public class MOPSSolver : ISolver, ICanSolve&lt;Model, Solution, IAabortable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual Basic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Class MOPSSolver _</td>
</tr>
<tr>
<td>Implements ISolver, ICanSolve(Of Model, Solution, IAabortable)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>public ref class MOPSSolver : ISolver, ICanSolve&lt;Model^, Solution^&gt;, IAabortable</td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
Optimization.Solver.MOPS.MOPSSolver
See Also

MOPSSolver Members
Optimization.Solver.MOPS Namespace
The **MOPSSolver** type exposes the following members.
### Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOPSSolver</td>
<td>Initializes a new instance of the MOPSSolver class</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Abort</strong></td>
<td></td>
</tr>
<tr>
<td><strong>ClearLastModel</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Solve</strong></td>
<td></td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td></td>
</tr>
<tr>
<td>IPObjectiveValue</td>
<td></td>
</tr>
<tr>
<td>IsAbortRequested</td>
<td></td>
</tr>
<tr>
<td>IsBusy</td>
<td></td>
</tr>
<tr>
<td>LPObjectiveValue</td>
<td></td>
</tr>
<tr>
<td>NumberOfInfeasibilities</td>
<td></td>
</tr>
<tr>
<td>NumberOfIPSolutions</td>
<td></td>
</tr>
<tr>
<td>NumberOfIterations</td>
<td></td>
</tr>
<tr>
<td>NumberOfNodes</td>
<td></td>
</tr>
<tr>
<td>ProfileName</td>
<td></td>
</tr>
</tbody>
</table>
See Also

MOPSSolver Class
Optimization.Solver.MOPS Namespace
Initializes a new instance of the **MOPSSolver** class

**Namespace:** [Optimization.Solver.MOPS](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#** | ```
public MOPSSolver(
    MOPSSolverConfiguration configuration
)
``` |
| **Visual Basic** | ```
Public Sub New (_
    configuration As MOPSSolverConfiguration _
)```
| **Visual C++** | ```
public:
MOPSSolver(              
    MOPSSolverConfiguration^ configuration
)``` |

### Parameters

*configuration*

Type: Optimization.Solver.MOPS.MOPSSolverConfiguration

See Also

MOPSSolver Class
Optimization.Solver.MOPS Namespace
The **MOPSSolver** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Abort</strong></td>
<td></td>
</tr>
<tr>
<td><strong>ClearLastModel</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is equal to the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.type">Type</a> of the current instance. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Solve</strong></td>
<td></td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="https://docs.microsoft.com/en-us/dotnet/api/system.string">String</a> that represents the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

MOPSSolver Class
Optimization.Solver.MOPS Namespace

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Method</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public void Abort()</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Sub Abort</code></td>
<td></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public: virtual void Abort() sealed</code></td>
<td></td>
</tr>
</tbody>
</table>

**Implements**

IAbortable.Abtort
See Also

MOPSSolver Class
Optimization.Solver.MOPS Namespace
Syntax

**C#**

```java
public void ClearLastModel()
```

**Visual Basic**

```vbnet
Public Sub ClearLastModel
```

**Visual C++**

```cpp
public:
virtual void ClearLastModel() sealed
```

**Implements**

`ISolver.ClearLastModel()`
See Also

MOPSSolver Class
Optimization.Solver.MOPS Namespace
Optimization Framework

MOPSSolver.Solve Method

MOPSSolver Class See Also Send Feedback


Namespace: Optimization.Solver.MOPS
Assembly: Optimization_Framework (in Optimization_Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public Solution Solve(
    Model model,
    IDictionary<string, double> variableValues
)
```

### Visual Basic

```vbnet
Public Function Solve (_
    model As Model, _
    variableValues As IDictionary(Of String, Double)
) As Solution
```

### Visual C++

```cpp
public:
    virtual Solution^ Solve(
        Model^ model,
        IDictionary<String^, double>^ variableValues
    ) sealed
```

## Parameters

- **model**
  Type: `Optimization.Model`

- **variableValues**
  Type: `System.Collections.Generic.IDictionary<string, double>`
Return Value

[Missing <returns> documentation for
]

Implements

ICanSolve(M, S).Solve(M, IDictionary(String, Double))
See Also

MOPSSolver Class
Optimization.Solver.MOPS Namespace
The **MOPSSolver** type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td></td>
</tr>
<tr>
<td>IPObjectiveValue</td>
<td></td>
</tr>
<tr>
<td>IsAbortRequested</td>
<td></td>
</tr>
<tr>
<td>IsBusy</td>
<td></td>
</tr>
<tr>
<td>LPObjectiveValue</td>
<td></td>
</tr>
<tr>
<td>NumberOfInfeasibilities</td>
<td></td>
</tr>
<tr>
<td>NumberOfIPSolutions</td>
<td></td>
</tr>
<tr>
<td>NumberOfIterations</td>
<td></td>
</tr>
<tr>
<td>NumberOfNodes</td>
<td></td>
</tr>
<tr>
<td>ProfileName</td>
<td></td>
</tr>
</tbody>
</table>
See Also

MOPSSolver Class
Optimization.Solver.MOPS Namespace

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th><strong>C#</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>public SolverConfiguration Configuration { get; set; }</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Visual Basic</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Property Configuration As SolverConfiguration Get Set</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Visual C++</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>public: virtual property SolverConfiguration^ Configuration : SolverConfiguration^ get () sealed; void set (SolverConfiguration^ value) sealed</td>
</tr>
</tbody>
</table>

**Implements**

ISolver.Configuration
See Also

MOPSSolver Class
Optimization.Solver.MOPS Namespace
[Missing <summary> documentation for "P:Optimization.Solver.MOPS.MOPSSolver.IObjectiveValue"]

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>public double IPObjectiveValue { get; set; }</td>
<td><strong>Public Property IPObjectiveValue As Double</strong>&lt;br&gt;Get&lt;br&gt;Set</td>
<td>public: double IPObjectiveValue {&lt;br&gt;    double get ();&lt;br&gt;    void set (double value);&lt;br&gt;}</td>
</tr>
</tbody>
</table>
See Also

MOPSSolver Class
Optimization.Solver.MOPS Namespace
MOPSSolver.IsAbortRequested Property

MOPSSolver Class See Also Send Feedback


Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public <code>bool</code> IsAbortRequested { get; set; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Property IsAbortRequested As <code>Boolean</code></td>
</tr>
<tr>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>Set</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: property <code>bool</code> IsAbortRequested {</td>
</tr>
<tr>
<td></td>
<td><code>bool</code> get ();</td>
</tr>
<tr>
<td></td>
<td>void set (<code>bool</code> value);</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>
See Also

MOPSSolver Class
Optimization.Solver.MOPS Namespace
MISSING <summary> documentation for
"P:Optimization.Solver.MOPS.MOPSSolver.IsBusy"

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public bool IsBusy { get; private set; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Property IsBusy As Boolean  Get Private Set</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: virtual property bool IsBusy { bool get () sealed; private: void set (bool value) sealed; }</td>
</tr>
</tbody>
</table>

### Implements

ISolver.IsBusy
See Also

MOPSSolver Class
Optimization.Solver.MOPS Namespace
[Missing <summary> documentation for "P:Optimization.Solver.MOPS.MOPSSolver.LPObjectiveValue"]

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th></th>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>public double LPObjectiveValue { get; set; }</td>
<td>Public Property LPObjectiveValue As Double Get Set</td>
<td>public: property double LPObjectiveValue { double get (); void set (double value);</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>
See Also

MOPSSolver Class
Optimization.Solver.MOPS Namespace
Optimization Framework

MOPSSolver.NumberOfInfeasibilities Property

MOPSSolver Class  See Also  Send Feedback

[Missing <summary> documentation for "P:Optimization.Solver.MOPS.MOPSSolver.NumberOfInfeasibilities"]

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

**C#**

```csharp
public int NumberOfInfeasiblities { get; set; }
```

**Visual Basic**

```vbnet
Public Property NumberOfInfeasiblities As Integer
    Get
        Set
```

**Visual C++**

```c++
public:
    property int NumberOfInfeasiblities {
        int get ();
        void set (int value);
    }
```
See Also

MOPSSolver Class
Optimization.Solver.MOPS Namespace
MOPSSolver.NumberOfIPSolutions Property

[Missing <summary> documentation for "P:Optimization.Solver.MOPS.MOPSSolver.NumberOfIPSolutions"]

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
</table>
| `public int NumberOfIPSolutions { get; set; }` | `Public Property NumberOfIPSolutions As Integer`<br>`Get`<br>`Set` | `public: int NumberOfIPSolutions {
  int get ();
  void set (int value);
};` |
See Also

MOPSSolver Class
Optimization.Solver.MOPS Namespace
Optimization Framework

MOPSSolver.NumberOfIterations Property

MOPSSolver Class  See Also  Send Feedback

[Missing <summary> documentation for "P:Optimization.Solver.MOPS.MOPSSolver.NumberOfIterations" ]

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public int NumberOfIterations { get; set; }</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Property NumberOfIterations As Integer</code></td>
</tr>
<tr>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>Set</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public: int NumberOfIterations {</code></td>
</tr>
<tr>
<td></td>
<td><code>int get ();</code></td>
</tr>
<tr>
<td></td>
<td><code>void set (int value);</code></td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>
See Also

MOPSSolver Class
Optimization.Solver.MOPS Namespace
MOPSSolver.NumberOfNodes Property

MOPSSolver Class  See Also  Send Feedback

[Missing <summary> documentation for "P:Optimization.Solver.MOPS.MOPSSolver.NumberOfNodes"]

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#**   | ```
public int NumberOfNodes { get; set; }
``` |
| **Visual Basic** | ```
Public Property NumberOfNodes As Integer
    Get
    Set
``` |
| **Visual C++** | ```
public:
    property int NumberOfNodes {
        int get ();
        void set (int value);
    }
``` |
See Also

- MOPSSolver Class
- Optimization.Solver.MOPS Namespace
MOPSSolver.ProfileName Property

[Missing <summary> documentation for "P:Optimization.Solver.MOPS.MOPSSolver.ProfileName"]

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
</tr>
<tr>
<td>public <code>string</code> ProfileName { get; set; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
</tr>
</tbody>
</table>
| Public Property ProfileName As `String`  
  Get  
  Set |
| **Visual C++** |
| public:  
  property `String`^ ProfileName {  
    `String`^ get ();  
    void set (`String`^ value);  
  } |
See Also

MOPSSolver Class
Optimization.Solver.MOPS Namespace
[Missing <summary> documentation for "T:Optimization.Solver.MOPS.MOPSSolverConfiguration"]

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
</tr>
<tr>
<td><code>public class MOPSSolverConfiguration : SolverConfiguration</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
</tr>
</tbody>
</table>
| `Public Class MOPSSolverConfiguration _
Inherits SolverConfiguration` |
| **Visual C++** |
| `public ref class MOPSSolverConfiguration : public SolverConfiguration` |
Inheritance Hierarchy

System.Object
Optimization.SolverConfiguration
Optimization.Solver.MOPS.MOPSSolverConfiguration
See Also

MOPSSolverConfiguration Members
Optimization.Solver.MOPS Namespace
The **MOPSSolverConfiguration** type exposes the following members.
### Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOPSSolverConfiguration()</td>
<td>Initializes a new instance of the MOPSSolverConfiguration class</td>
</tr>
<tr>
<td>MOPSSolverConfiguration(String)</td>
<td>Initializes a new instance of the MOPSSolverConfiguration class</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CallbackEndpoint</td>
<td>This endpoint will be used to enable communication between solvers (Inherited from <a href="#">SolverConfiguration</a>.)</td>
</tr>
<tr>
<td>ConfigFile</td>
<td>(Inherited from <a href="#">SolverConfiguration</a>.)</td>
</tr>
<tr>
<td>GetGlobalIncumbent</td>
<td>(Inherited from <a href="#">SolverConfiguration</a>.)</td>
</tr>
<tr>
<td>NewBestBoundFound</td>
<td>This action is executed whenever a new best bound is found (Inherited from <a href="#">SolverConfiguration</a>.)</td>
</tr>
<tr>
<td>NewIncumbentFound</td>
<td>This action is executed whenever a new incumbent is found (Inherited from <a href="#">SolverConfiguration</a>.)</td>
</tr>
<tr>
<td>PartialProblemsNeeded</td>
<td>A method that decides if a partial problem should be created. (Inherited from <a href="#">SolverConfiguration</a>.)</td>
</tr>
<tr>
<td>SendPartialProblems</td>
<td>A method that sends partial problems over the wire (Inherited from <a href="#">SolverConfiguration</a>.)</td>
</tr>
<tr>
<td>UseHeuristicCallback</td>
<td>Indicates if the solver should activate its provided HeuristicCallback This will make the solver use injected solutions from other solvers (Inherited from <a href="#">SolverConfiguration</a>.)</td>
</tr>
</tbody>
</table>
See Also

MOPSSolverConfiguration Class
Optimization.Solver.MOPS Namespace
Optimization Framework

MOPSSolverConfiguration Constructor

MOPSSolverConfiguration Class  See Also  Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOPSSolverConfiguration()</td>
<td>Initializes a new instance of the MOPSSolverConfiguration class</td>
</tr>
<tr>
<td>MOPSSolverConfiguration(String)</td>
<td>Initializes a new instance of the MOPSSolverConfiguration class</td>
</tr>
</tbody>
</table>
See Also

MOPSSolverConfiguration Class
MOPSSolverConfiguration Members
Optimization.Solver.MOPS Namespace
Initializes a new instance of the MOPSSolverConfiguration class

**Namespace:** Optimization.Solver.MOPS  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
<tr>
<td>public MOPSSolverConfiguration()</td>
<td></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td></td>
</tr>
<tr>
<td>Public Sub New</td>
<td></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
</tr>
<tr>
<td>public: MOPSSolverConfiguration()</td>
<td></td>
</tr>
</tbody>
</table>
See Also

MOPSSolverConfiguration Class
MOPSSolverConfiguration Overload
Optimization.Solver.MOPS Namespace
Initializes a new instance of the MOPSSolverConfiguration class

**Namespace:** Optimization.Solver.MOPS  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>public MOPSSolverConfiguration(</code></td>
<td><code>Public Sub New (_</code></td>
<td><code>public:</code></td>
</tr>
<tr>
<td><code>    string profile</code></td>
<td><code>    profile As String _</code></td>
<td><code>MOPSSolverConfiguration(</code></td>
</tr>
<tr>
<td><code>)</code></td>
<td><code>)</code></td>
<td><code>String^ profile</code></td>
</tr>
</tbody>
</table>

### Parameters

`profile`

Type: `System.String`

[Missing <param name="profile"/> documentation for "M:Optimization.Solver.MOPS.MOPSSolverConfiguration.#ctor(System."
See Also

MOPSSolverConfiguration Class
MOPSSolverConfiguration Overload
Optimization.Solver.MOPS Namespace
The **MOPSSolverConfiguration** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

MOPSSolverConfiguration Class
Optimization.Solver.MOPS Namespace
The **MOPSSolverConfiguration** type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CallbackEndpoint</strong></td>
<td>This endpoint will be used to enable communication between solvers (Inherited from <a href="#">SolverConfiguration</a>).</td>
</tr>
<tr>
<td><strong>ConfigFile</strong></td>
<td></td>
</tr>
<tr>
<td><strong>GetGlobalIncumbent</strong></td>
<td>(Inherited from <a href="#">SolverConfiguration</a>).</td>
</tr>
<tr>
<td><strong>NewBestBoundFound</strong></td>
<td>This action is executed whenever a new best bound is found (Inherited from <a href="#">SolverConfiguration</a>).</td>
</tr>
<tr>
<td><strong>NewIncumbentFound</strong></td>
<td>This action is executed whenever a new incumbent is found (Inherited from <a href="#">SolverConfiguration</a>).</td>
</tr>
<tr>
<td><strong>PartialProblemsNeeded</strong></td>
<td>A method that decides if a partial problem should be created. (Inherited from <a href="#">SolverConfiguration</a>).</td>
</tr>
<tr>
<td><strong>SendPartialProblems</strong></td>
<td>A method that sends partial problems over the wire (Inherited from <a href="#">SolverConfiguration</a>).</td>
</tr>
<tr>
<td><strong>UseHeuristicCallback</strong></td>
<td>Indicates if the solver should activate its provided HeuristicCallback This will make the solver use injected solutions from other solvers (Inherited from <a href="#">SolverConfiguration</a>).</td>
</tr>
</tbody>
</table>
See Also

MOPSSolverConfiguration Class
Optimization.Solver.MOPS Namespace
MOPSSolverConfiguration.ConfigFile Property

MOPSSolverConfiguration Class  See Also  Send Feedback

[Missing <summary> documentation for
"P:Optimization.Solver.MOPS.MOPSSolverConfiguration.ConfigFile"]

Namespace: Optimization.Solver.MOPS
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
</tr>
<tr>
<td>public string ConfigFile { get; set; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
</tr>
<tr>
<td>Public Property ConfigFile As String</td>
</tr>
<tr>
<td>Get</td>
</tr>
<tr>
<td>Set</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
</tr>
<tr>
<td>public: property String^ ConfigFile {</td>
</tr>
<tr>
<td>String^ get ();</td>
</tr>
<tr>
<td>void set (String^ value);</td>
</tr>
<tr>
<td>}</td>
</tr>
</tbody>
</table>
See Also

MOPSSolverConfiguration Class
Optimization.Solver.MOPS Namespace
Namespace for everything that lets you use Unity as an IOC with the framework
### Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>DefaultHostFactory</code></td>
<td>This class is needed whenever you want to use the UnityServiceHostFactory with IIS</td>
</tr>
<tr>
<td><code>Unity</code></td>
<td>A class that defines functions for unity.</td>
</tr>
<tr>
<td><code>UnityInstanceProvider</code></td>
<td>An instance provider for unity</td>
</tr>
<tr>
<td><code>UnityServiceBehavior</code></td>
<td>A service behavior for a unity service host</td>
</tr>
<tr>
<td><code>UnityServiceHost</code></td>
<td>A ServiceHost that uses Unity for dependency injection</td>
</tr>
<tr>
<td><code>UnityServiceHostFactory</code></td>
<td>Allows to create unity service hosts.</td>
</tr>
<tr>
<td><code>WcfUnityConfiguration</code></td>
<td>Allows configuration of a unity container in a WCF context</td>
</tr>
</tbody>
</table>
### Interfaces

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>IUnityConfiguration</code></td>
<td>Exposes a method to configure the unity container</td>
</tr>
</tbody>
</table>
This class is needed whenever you want to use the UnityServiceHostFactory with IIS

**Namespace**: Optimization.Unity  
**Assembly**: Optimization.Framework (in Optimization.Framework.dll)  
**Version**: 0.3.1036.0
## Syntax

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>```</td>
<td>public class DefaultHostFactory : UnityServiceHostFactory ```</td>
<td>```</td>
<td>```</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>```</td>
<td>Public Class DefaultHostFactory Inherits UnityServiceHostFactory ```</td>
<td>```</td>
<td>```</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>```</td>
<td>public ref class DefaultHostFactory : public UnityServiceHostFactory ```</td>
<td>```</td>
<td>```</td>
</tr>
</tbody>
</table>


Inheritance Hierarchy

System.Object
System.ServiceModel.Activation.ServiceHostFactoryBase
Optimization.Unity.UnityServiceHostFactory
Optimization.Unity.DefaultHostFactory
See Also

DefaultHostFactory Members
Optimization.Unity Namespace
The `DefaultHostFactory` type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DefaultHostFactory</td>
<td>Initializes a new instance of the DefaultHostFactory class.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CreateServiceHost</td>
<td>When overridden in a derived class, creates a <a href="#">ServiceHostBase</a> with a specific base address using custom initiation data. (Inherited from <a href="#">UnityServiceHostFactory</a>.)</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

DefaultHostFactory Class
Optimization.Unity Namespace
Optimization Framework

DefaultHostFactory Constructor

DefaultHostFactory Class See Also Send Feedback

Initializes a new instance of the DefaultHostFactory class.

Namespace: Optimization.Unity
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public DefaultHostFactory()</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Sub New</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public: DefaultHostFactory()</code></td>
</tr>
</tbody>
</table>
See Also

DefaultHostFactory Class
Optimization.Unity Namespace
The `DefaultHostFactory` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateServiceHost</td>
<td>When overridden in a derived class, creates a <code>ServiceHostBase</code> with a specific base address using custom initiation data. (Inherited from <code>UnityServiceHostFactory</code>.)</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
</tbody>
</table>
See Also

* DefaultHostFactory Class
* Optimization.Unity Namespace
Exposes a method to configure the unity container

**Namespace:** Optimization.Unity

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public interface IUnityConfiguration</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Interface IUnityConfiguration</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public interface class IUnityConfiguration</code></td>
</tr>
</tbody>
</table>
See Also

IUnityConfiguration Members
Optimization.Unity Namespace
The **IUnityConfiguration** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure</td>
<td>Configures the specified container.</td>
</tr>
</tbody>
</table>
See Also

IUnityConfiguration Interface
Optimization.Unity Namespace
The **IUnityConfiguration** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure</td>
<td>Configures the specified container.</td>
</tr>
</tbody>
</table>
**See Also**

- IUnityConfiguration Interface
- Optimization.Unity Namespace
IUnityConfiguration. Configure Method

Configures the specified container.

Namespace: Optimization.Unity
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
**Syntax**

**C#**

```csharp
void Configure(
    IUnityContainer container
)
```

**Visual Basic**

```vbnet
Sub Configure ( _
    container As IUnityContainer _
)
```

**Visual C++**

```cpp
void Configure(
    IUnityContainer^ container
)
```

**Parameters**

*container*

- Type: `IUnityContainer`
- The container.
See Also

IUnityConfiguration Interface
Optimization.Unity Namespace
A class that defines functions for unity.

**Namespace:** Optimization.Unity

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
<th>C#</th>
<th>Visual Basic</th>
<th>Visual C++</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>public static class Unity</td>
<td>Public NotInheritable Class</td>
<td>public ref class Unity abstract sealed</td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
Optimization.Unity.Unity
See Also

Unity Members
Optimization.Unity Namespace
The **Unity** type exposes the following members.
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure</td>
<td>Configures the specified configuration.</td>
</tr>
<tr>
<td>Resolve(T)</td>
<td>Resolves this instance.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>Current</td>
<td>Gets the current unity container</td>
</tr>
</tbody>
</table>
See Also

Unity Class
Optimization.Unity Namespace
The **Unity** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure</td>
<td>Configures the specified configuration.</td>
</tr>
<tr>
<td>Resolve(T)</td>
<td>Resolves this instance.</td>
</tr>
</tbody>
</table>
See Also

Unity Class
Optimization.Unity Namespace
Configures the specified configuration.

**Namespace:** [Optimization.Unity](https://example.com)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public static void Configure(
    IUnityConfiguration configuration
)
```

### Visual Basic

```vbnet
Public Shared Sub Configure (_
    configuration As IUnityConfiguration _
)
```

### Visual C++

```cpp
public:
static void Configure(
    IUnityConfiguration^ configuration
)
```

## Parameters

- **configuration**
  - Type: [Optimization.Unity.IUnityConfiguration](#)
  - The configuration.
**See Also**

Unity Class
Optimization.Unity Namespace
Optimization Framework

Unity.Resolve(T) Method

Namespace: Optimization.Unity
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0

Resolves this instance.
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public static T Resolve&lt;T&gt;()</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Shared Function Resolve(Of T) As T</code></td>
</tr>
</tbody>
</table>
| **Visual C++** | `public:
generic<typename T>
static T Resolve()`          |
Type Parameters

$T$
The type to resolve

Return Value
The resolved instance
See Also

Unity Class
Optimization.Unity Namespace
The **Unity** type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current</td>
<td>Gets the current unity container</td>
</tr>
</tbody>
</table>
See Also

Unity Class
Optimization.Unity Namespace
Optimization Framework

Unity.Current Property

Unity Class See Also Send Feedback

Gets the current unity container

Namespace: Optimization.Unity
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

**C#**

```cshar
public static IUnityContainer Current { get; }
```

**Visual Basic**

```vbnet
Public Shared ReadOnly Property Current As IUnityContainer
    Get
```

**Visual C++**

```cpp
public:
    static property IUnityContainer^ Current
    {
        IUnityContainer^ get ();
    }
```

**Field Value**

The current unity container
See Also

Unity Class
Optimization.Unity Namespace
An instance provider for unity

**Namespace:** [Optimization.Unity](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
</tr>
<tr>
<td><code>public class UnityInstanceProvider : IInstanceProvider</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
</tr>
<tr>
<td><code>Public Class UnityInstanceProvider _ Implements IInstanceProvider</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
</tr>
<tr>
<td><code>public ref class UnityInstanceProvider : IInstanceProvider</code></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
Optimization.Unity.UnityInstanceProvider
See Also

UnityInstanceProvider Members
Optimization.Unity Namespace
The `UnityInstanceProvider` type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>UnityInstanceProvider()</code></td>
<td>Initializes a new instance of the <code>UnityInstanceProvider</code> class.</td>
</tr>
<tr>
<td><code>UnityInstanceProvider(Type)</code></td>
<td>Initializes a new instance of the <code>UnityInstanceProvider</code> class.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals</td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup</td>
</tr>
<tr>
<td></td>
<td>operations before the <code>Object</code> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>GetInstance(InstanceContext)</td>
<td>Returns a service object given the specified <code>InstanceContext</code> object.</td>
</tr>
<tr>
<td>GetInstance(InstanceContext, Message)</td>
<td>Returns a service object given the specified <code>InstanceContext</code> object.</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <code>Type</code> of the current instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <code>Object</code>.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>ReleaseInstance</td>
<td>Called when an <code>InstanceContext</code> object recycles a service object.</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Container</td>
<td>Gets or sets the unity container.</td>
</tr>
<tr>
<td>ServiceType</td>
<td>Gets or sets the type of the service.</td>
</tr>
</tbody>
</table>
See Also

UnityInstanceProvider Class
Optimization.Unity Namespace
Optimization Framework

UnityInstanceProvider Constructor

UnityInstanceProvider Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UnityInstanceProvider()</td>
<td>Initializes a new instance of the UnityInstanceProvider class.</td>
</tr>
<tr>
<td>UnityInstanceProvider(Type)</td>
<td>Initializes a new instance of the UnityInstanceProvider class.</td>
</tr>
</tbody>
</table>
See Also

UnityInstanceProvider Class
UnityInstanceProvider Members
Optimization.Unity Namespace
Initializes a new instance of the `UnityInstanceProvider` class.

**Namespace:** Optimization.Unity  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
## Syntax

### C#

```
public UnityInstanceProvider()
```

### Visual Basic

```
Public Sub New
```

### Visual C++

```
public:
UnityInstanceProvider()
```
See Also

UnityInstanceProvider Class
UnityInstanceProvider Overload
Optimization.Unity Namespace
Optimization Framework

UnityInstanceProvider Constructor (Type)

Initializes a new instance of the `UnityInstanceProvider` class.

**Namespace:** Optimization.Unity

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public UnityInstanceProvider(
    Type type
)
```

### Visual Basic

```vbnet
Public Sub New (_
    type As Type _
)
```

### Visual C++

```cpp
public:
UnityInstanceProvider(
    Type^ type
)
```

## Parameters

- **type**
  - Type: `System.Type`
  - The type.
See Also

UnityInstanceProvider Class
UnityInstanceProvider Overload
Optimization.Unity Namespace
The **UnityInstanceProvider** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="Object">Object</a> is equal to the current <a href="Object">Object</a>. (Inherited from <a href="Object">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="Object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="Object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="Object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="Object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetInstance(InstanceContext)</strong></td>
<td>Returns a service object given the specified <a href="InstanceContext">InstanceContext</a> object.</td>
</tr>
<tr>
<td><strong>GetInstance(InstanceContext, Message)</strong></td>
<td>Returns a service object given the specified <a href="InstanceContext">InstanceContext</a> object.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="Type">Type</a> of the current instance. (Inherited from <a href="Object">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="Object">Object</a>. (Inherited from <a href="Object">Object</a>.)</td>
</tr>
<tr>
<td><strong>ReleaseInstance</strong></td>
<td>Called when an <a href="InstanceContext">InstanceContext</a> object recycles a service object.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="String">String</a> that represents the current <a href="Object">Object</a>. (Inherited from <a href="Object">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

UnityInstanceProvider Class
Optimization.Unity Namespace
Optimization Framework

UnityInstanceProvider.GetInstance Method

UnityInstanceProvider Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>GetInstance(InstanceContext)</code></td>
<td>Returns a service object given the specified <code>InstanceContext</code> object.</td>
</tr>
<tr>
<td><code>GetInstance(InstanceContext, Message)</code></td>
<td>Returns a service object given the specified <code>InstanceContext</code> object.</td>
</tr>
</tbody>
</table>
See Also

UnityInstanceProvider Class
UnityInstanceProvider Members
Optimization.Unity Namespace
Returns a service object given the specified `InstanceContext` object.

**Namespace:** Optimization.Unity

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

**C#**

```csharp
public Object GetInstance(InstanceContext instanceContext)
```

**Visual Basic**

```vbnet
Public Function GetInstance ( _
    instanceContext As InstanceContext _
) As Object
```

**Visual C++**

```cpp
public:
    virtual Object^ GetInstance(InstanceContext^ instanceContext)
```

## Parameters

*instanceContext*

Type: `System.ServiceModel.InstanceContext`

The current `InstanceContext` object.

## Return Value

A user-defined service object.

## Implements

`IInstanceProvider, GetInstance(InstanceContext)`
See Also

UnityInstanceProvider Class
GetInstance Overload
Optimization.Unity Namespace
Returns a service object given the specified `InstanceContext` object.

**Namespace:** Optimization.Unity  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

**C#**

```csharp
public Object GetInstance(
    InstanceContext instanceContext, 
    Message message
)
```

**Visual Basic**

```vbnet
Public Function GetInstance ( _
    instanceContext As InstanceContext, _
    message As Message _
) As Object
```

**Visual C++**

```cpp
public:
    virtual Object^ GetInstance(
        InstanceContext^ instanceContext, 
        Message^ message
    )
```

### Parameters

- **instanceContext**
  Type: `System.ServiceModel.InstanceContext`
  The current `InstanceContext` object.

- **message**
  Type: `System.ServiceModel.Channels.Message`
  The message that triggered the creation of a service object.

### Return Value
The service object.

**Implements**

`IInstanceProvider.GetInstance(InstanceContext, Message)`
See Also

UnityInstanceProvider Class
GetInstance Overload
Optimization.Unity Namespace
Called when an `InstanceContext` object recycles a service object.

**Namespace:** Optimization.Unity

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public void ReleaseInstance(
    InstanceContext instanceContext,
    Object instance
)
```

Visual Basic

```vbnet
Public Sub ReleaseInstance (_
    instanceContext As InstanceContext, _
    instance As Object _
)
```

Visual C++

```c++
public:
    virtual void ReleaseInstance(
        InstanceContext& instanceContext,
        Object& instance
    )
```

sealed

Parameters

`instanceContext`
Type: `System.ServiceModel.InstanceContext`
The service's instance context.

`instance`
Type: `System.Object`
The service object to be recycled.

Implements
IInstanceProvider.ReleaseInstance(InstanceContext, Object)
See Also

UnityInstanceProvider Class
Optimization.Unity Namespace
The `UnityInstanceProvider` type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Container</td>
<td>Gets or sets the unity container.</td>
</tr>
<tr>
<td>ServiceType</td>
<td>Gets or sets the type of the service.</td>
</tr>
</tbody>
</table>
See Also

UnityInstanceProvider Class
Optimization.Unity Namespace
Optimization Framework

UnityInstanceProvider.Container Property

Gets or sets the unity container.

**Namespace:** Optimization.Unity

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public IUnityContainer Container { get; set; }
```

### Visual Basic

```vbnet
Public Property Container As IUnityContainer
    Get
    Set
```

### Visual C++

```cpp
public:
    property IUnityContainer^ Container {
        IUnityContainer^ get ();
        void set (IUnityContainer^ value);
    }
```

### Field Value

The container.
See Also

UnityInstanceProvider Class
Optimization.Unity Namespace
Gets or sets the type of the service.

**Namespace:** [Optimization.Unity](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td>public Type ServiceType { get; set; }</td>
</tr>
<tr>
<td>Visual Basic</td>
<td>Public Property ServiceType As Type</td>
</tr>
<tr>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>Set</td>
</tr>
<tr>
<td>Visual C++</td>
<td>public: property Type^ ServiceType {</td>
</tr>
<tr>
<td></td>
<td>Type^ get ();</td>
</tr>
<tr>
<td></td>
<td>void set (Type^ value);</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

**Field Value**

The type of the service.
See Also

UnityInstanceProvider Class
Optimization.Unity Namespace
A service behavior for a unity service host

Namespace: Optimization.Unity
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public class UnityServiceBehavior : IServiceBehavior</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Class UnityServiceBehavior _ Implements IServiceBehavior</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public ref class UnityServiceBehavior : IServiceBehavior</code></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
Optimization.UnityEngine.ServiceBehavior
See Also

UnityServiceBehavior Members
Optimization.Unity Namespace
The **UnityServiceBehavior** type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>UnityEngineServiceBehavior()</code></td>
<td>Initializes a new instance of the <code>UnityEngineServiceBehavior</code> class.</td>
</tr>
<tr>
<td><code>UnityEngineServiceBehavior(IUnityContainer)</code></td>
<td>Initializes a new instance of the <code>UnityEngineServiceBehavior</code> class.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AddBindingParameters</strong></td>
<td>Provides the ability to pass custom data to binding elements to support the contract implementation.</td>
</tr>
<tr>
<td><strong>AddToHost</strong></td>
<td>Adds this service behavior to the specified host.</td>
</tr>
<tr>
<td><strong>ApplyDispatchBehavior</strong></td>
<td>Provides the ability to change run-time property values or insert custom extension objects such as error handlers, message or parameter interceptors, security extensions, and other custom extension objects.</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <strong>Type</strong> of the current instance. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>Validate</strong></td>
<td>Provides the ability to inspect the service host and the service description to confirm that the service can run.</td>
</tr>
</tbody>
</table>
successfully.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InstanceProvider</td>
<td>Gets or sets the instance provider.</td>
</tr>
</tbody>
</table>
See Also

UnityServiceBehavior Class
Optimization.Unity Namespace
Optimization Framework

UnityServiceBehavior Constructor

UnityServiceBehavior Class See Also Send Feedback
Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UnityServiceBehavior()</td>
<td>Initializes a new instance of the UnityServiceBehavior class.</td>
</tr>
<tr>
<td>UnityServiceBehavior(IUnityContainer)</td>
<td>Initializes a new instance of the UnityServiceBehavior class.</td>
</tr>
</tbody>
</table>
See Also

UnityServiceBehavior Class
UnityServiceBehavior Members
Optimization.Unity Namespace
Optimization Framework

UnityServiceBehavior Constructor

**UnityServiceBehavior Class** See Also Send Feedback

Initializes a new instance of the **UnityServiceBehavior** class.

**Namespace:** [Optimization.Unity](https://www.optimization.com/unity)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public UnityServiceBehavior()</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Sub New</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: UnityServiceBehavior()</td>
</tr>
</tbody>
</table>
See Also

UnityServiceBehavior Class
UnityServiceBehavior Overload
Optimization.Unity Namespace
UnityServiceBehavior Constructor ()

Initializes a new instance of the UnityServiceBehavior class.

**Namespace:** Optimization.Unity

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public UnityServiceBehavior(IUnityContainer unity)</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Sub New ( _ unity As IUnityContainer _ )</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>public: UnityServiceBehavior(IUnityContainer^ unity)</td>
</tr>
</tbody>
</table>

### Parameters

- **unity**  
  Type: **IUnityContainer**  
  The unity.
See Also

UnityServiceBehavior Class
UnityServiceBehavior Overload
Optimization.Unity Namespace
The **UnityServiceBehavior** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddBindingParameters</td>
<td>Provides the ability to pass custom data to binding elements to support the contract implementation.</td>
</tr>
<tr>
<td>AddToHost</td>
<td>Adds this service behavior to the specified host.</td>
</tr>
<tr>
<td>ApplyDispatchBehavior</td>
<td>Provides the ability to change run-time property values or insert custom extension objects such as error handlers, message or parameter interceptors, security extensions, and other custom extension objects.</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified Object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a String that represents the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>Validate</td>
<td>Provides the ability to inspect the service host and the service description to confirm that the service can run</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
See Also

UnityServiceBehavior Class
Optimization.Unity Namespace
Optimization Framework

UnityServiceBehavior.AddBindingParameters Method

Provides the ability to pass custom data to binding elements to support the contract implementation.

**Namespace:** Optimization.Unity

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
Syntax

C#

```csharp
public void AddBindingParameters(
    ServiceDescription serviceDescription,
    ServiceHostBase serviceHostBase,
    Collection<ServiceEndpoint> endpoints,
    BindingParameterCollection bindingParameters
)
```

Visual Basic

```vbnet
Public Sub AddBindingParameters ( _
    serviceDescription As ServiceDescription, _
    serviceHostBase As ServiceHostBase, _
    endpoints As Collection(Of ServiceEndpoint), _
    bindingParameters As BindingParameterCollection
)
```

Visual C++

```cpp
public:
    virtual void AddBindingParameters(
        ServiceDescription^ serviceDescription,
        ServiceHostBase^ serviceHostBase,
        Collection<ServiceEndpoint^>^ endpoints,
        BindingParameterCollection^ bindingParameters
    ) sealed
```

Parameters

`serviceDescription`
Type: [System.ServiceModel.Description.ServiceDescription](https://docs.microsoft.com/en-us/dotnet/api/system.servicemodel.description.servicedescription)
The service description of the service.
serviceHostBase
Type: System.ServiceModel.ServiceHostBase
The host of the service.

endpoints
The service endpoints.

bindingParameters
Type: System.ServiceModel.Channels.BindingParameterCollection
Custom objects to which binding elements have access.

Implements
IServiceBehavior.AddBindingParameters(ServiceDescription, ServiceHostBase, Collection<ServiceEndpoint>, BindingParameterCollection)
See Also

UnityServiceBehavior Class
Optimization.Unity Namespace
Optimization Framework

UnityServiceBehavior.AddToHost Method

UnityServiceBehavior Class See Also Send Feedback

Adds this service behavior to the specified host.

Namespace: Optimization.Unity
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public void AddToHost(
    ServiceHost host
)
```

#### Visual Basic

```vbnet
Public Sub AddToHost (_
    host As ServiceHost _
)
```

#### Visual C++

```cpp
public:
void AddToHost(
    ServiceHost^ host
)
```

### Parameters

*host*

Type: `System.ServiceModel.ServiceHost`

The host.
### Contracts

<table>
<thead>
<tr>
<th>Requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>host!=null</td>
</tr>
<tr>
<td>host.Description!=null</td>
</tr>
<tr>
<td>host.Description.Behaviors!=null</td>
</tr>
</tbody>
</table>

[Learn more about contracts](#)
See Also

UnityServiceBehavior Class
Optimization.Unity Namespace
Optimization Framework

UnityServiceBehavior.ApplyDispatchBehavior Method

Provides the ability to change run-time property values or insert custom extension objects such as error handlers, message or parameter interceptors, security extensions, and other custom extension objects.

**Namespace:** [Optimization.Unity](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public void ApplyDispatchBehavior(
    ServiceDescription serviceDescription,
    ServiceHostBase serviceHostBase
)
```

### Visual Basic

```vbnet
Public Sub ApplyDispatchBehavior (_
    serviceDescription As ServiceDescription, _
    serviceHostBase As ServiceHostBase _
)
```

### Visual C++

```cpp
public:
    virtual void ApplyDispatchBehavior(
        ServiceDescription^ serviceDescription,
        ServiceHostBase^ serviceHostBase
    )
```

## Parameters

**serviceDescription**

Type: System.ServiceModel.Description.ServiceDescription

The service description.

**serviceHostBase**

Type: System.ServiceModel.ServiceHostBase

The host that is currently being built.

## Implements
IServiceBehavior.ApplyDispatchBehavior(ServiceDescription, ServiceHostBase)
See Also

UnityServiceBehavior Class
Optimization.Unity Namespace
Provides the ability to inspect the service host and the service description to confirm that the service can run successfully.

**Namespace:** Optimization.Unity  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll) **Version:** 0.3.1036.0
Syntax

**C#**

```csharp
public void Validate(
    ServiceDescription serviceDescription,
    ServiceHostBase serviceHostBase
)
```

**Visual Basic**

```vbnet
Public Sub Validate (_
    serviceDescription As ServiceDescription,
    serviceHostBase As ServiceHostBase _
)
```

**Visual C++**

```cpp
public:
    virtual void Validate(
        ServiceDescription^ serviceDescription,
        ServiceHostBase^ serviceHostBase
    ) sealed
```

**Parameters**

*serviceDescription*

Type: System.ServiceModel.Description.ServiceDescription

The service description.

*serviceHostBase*

Type: System.ServiceModel.ServiceHostBase

The service host that is currently being constructed.

**Implements**
IServiceBehavior.Validate(ServiceDescription, ServiceHostBase)
See Also

UnityServiceBehavior Class
Optimization.Unity Namespace
The **UnityServiceBehavior** type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InstanceProvider</td>
<td>Gets or sets the instance provider.</td>
</tr>
</tbody>
</table>
See Also

UnityServiceBehavior Class
Optimization.Unity Namespace
Optimization Framework

UnityServiceBehavior.InstanceProvider Property

See Also Send Feedback

Gets or sets the instance provider.

**Namespace:** [Optimization.Unity](#)  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>public <code>UnityInstanceProvider</code> InstanceProvider { get; }</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Public Property <code>InstanceProvider</code> As <code>UnityInstanceProvider</code>&lt;br&gt;Get&lt;br&gt;Set</td>
</tr>
</tbody>
</table>
| **Visual C++** | public:<br>property `UnityInstanceProvider`^{ ^} InstanceProvider {<br>UnityInstanceProvider^{ ^} get ();<br>void set (UnityInstanceProvider^{ ^} value);<br>}

### Field Value

The instance provider.
See Also

UnityServiceBehavior Class
Optimization.Unity Namespace
A ServiceHost that uses Unity for dependency injection

**Namespace:** [Optimization.Unity](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public class UnityServiceHost : ServiceHost
```

### Visual Basic

```vbnet
Public Class UnityServiceHost _
    Inherits ServiceHost
```

### Visual C++

```cpp
public ref class UnityServiceHost : public ServiceHost
```
Inheritance Hierarchy

- `System.Object`
- `System.ServiceModel.ServiceHostBase`
- `System.ServiceModel.ServiceHost`
- `Optimization.Unity.UnityServiceHost`
See Also

UnityServiceHost Members
Optimization.Unity Namespace
The **UnityServiceHost** type exposes the following members.
### Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>UnityServiceHost()</code></td>
<td>Initializes a new instance of the <code>UnityServiceHost</code> class.</td>
</tr>
<tr>
<td><code>UnityServiceHost(Type, Uri[])</code></td>
<td>Initializes a new instance of the <code>UnityServiceHost</code> class.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Abort</td>
<td>Causes a communication object to transition immediately from its current state into the closing state. (Inherited from CommunicationObject)</td>
</tr>
<tr>
<td>AddBaseAddress</td>
<td>Adds a base address to the service host.</td>
</tr>
<tr>
<td>AddDefaultEndpoints</td>
<td>Adds service endpoints for each address in each contract found in the service host with the default binding. (Inherited from ServiceHostBase)</td>
</tr>
<tr>
<td>AddServiceEndpoint(ServiceEndpoint)</td>
<td>Adds the specified service endpoint to the hosted service. (Inherited from ServiceHost)</td>
</tr>
<tr>
<td>AddServiceEndpoint(Type, Binding, String)</td>
<td>Adds a service endpoint to the hosted service with a specified contract, binding, and endpoint address. (Inherited from ServiceHostBase)</td>
</tr>
<tr>
<td>AddServiceEndpoint(Type, Binding, Uri)</td>
<td>Adds a service endpoint to the hosted service with a specified contract, binding, and URI that contains the endpoint address. (Inherited from ServiceHost)</td>
</tr>
<tr>
<td>AddServiceEndpoint(String, Binding, String)</td>
<td>Adds a service endpoint to the hosted service with a specified contract, binding, and endpoint address. (Inherited from ServiceHostBase)</td>
</tr>
<tr>
<td>AddServiceEndpoint(String, Binding, Uri)</td>
<td>Adds a service endpoint to the hosted service with a specified contract, binding, and a URI that contains the endpoint address. (Inherited from ServiceHost)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td><strong>AddServiceEndpoint</strong>(Type, Binding, String, Uri)</td>
<td>Adds a service endpoint to the hosted service with a specified contract, binding, an endpoint address, and a URI on which the service listens. (Inherited from ServiceHostBase)</td>
</tr>
<tr>
<td><strong>AddServiceEndpoint</strong>(Type, Binding, Uri, Uri)</td>
<td>Adds a service endpoint to the hosted service with a specified contract, binding, a URI that contains the endpoint address, and a URI on which the service listens. (Inherited from ServiceHost)</td>
</tr>
<tr>
<td><strong>AddServiceEndpoint</strong>(String, Binding, String, Uri)</td>
<td>Adds a service endpoint to the hosted service with a specified contract, binding, endpoint address and URI that contains the address at which it listens. (Inherited from ServiceHostBase)</td>
</tr>
<tr>
<td><strong>AddServiceEndpoint</strong>(String, Binding, Uri, Uri)</td>
<td>Adds a service endpoint to the hosted service with the specified contract, binding, and URIs that contain the endpoint and listening addresses. (Inherited from ServiceHostBase)</td>
</tr>
<tr>
<td><strong>ApplyConfiguration</strong></td>
<td>Loads the service description information from the configuration file and applies it to the runtime being constructed. (Inherited from ServiceHostBase)</td>
</tr>
<tr>
<td><strong>BeginClose</strong>(AsyncCallback, Object)</td>
<td>Begins an asynchronous operation to close a communication object. (Inherited from CommunicationObject)</td>
</tr>
<tr>
<td><strong>BeginClose</strong>(TimeSpan, AsyncCallback, Object)</td>
<td>Begins an asynchronous operation to close a communication object with a specified timeout. (Inherited from CommunicationObject)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| **BeginOpen(AsyncCallback, Object)** | Begins an asynchronous operation to open a communication object. (Inherited from **CommunicationObject**).
| **BeginOpen(TimeSpan, AsyncCallback, Object)** | Begins an asynchronous operation to open a communication object within a specified interval of time. (Inherited from **CommunicationObject**).
| **Close()** | Causes a communication object to transition from its current state into the closed state. (Inherited from **CommunicationObject**).
| **Close(TimeSpan)** | Causes a communication object to transition from its current state into the closed state within a specified interval of time. (Inherited from **CommunicationObject**).
| **CreateDescription** | Creates a description of the service hosted. (Inherited from **Service**).
| **EndClose** | Completes an asynchronous operation to close a communication object. (Inherited from **CommunicationObject**).
| **EndOpen** | Completes an asynchronous operation to open a communication object. (Inherited from **CommunicationObject**).
| **Equals** | Determines whether the specified **Object** is equal to the current object. (Inherited from **Object**).
| **Fault** | Causes a communication object to transition from its current state into the faulted state.
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>Inherited From</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection.</td>
<td>CommunicationObject</td>
</tr>
<tr>
<td><strong>GetCommunicationObjectType</strong></td>
<td>Gets the type of communication object.</td>
<td>Object</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type.</td>
<td>Object</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the Type of the current instance.</td>
<td>Object</td>
</tr>
<tr>
<td><strong>IncrementManualFlowControlLimit</strong></td>
<td>Increases the limit on the flow rate of messages to the hosted service by a specified increment.</td>
<td>ServiceHostBase</td>
</tr>
<tr>
<td><strong>InitializeDescription(UriSchemeKeyedCollection)</strong></td>
<td>Creates and initializes the service host with the contract and service descriptions.</td>
<td>ServiceHostBase</td>
</tr>
<tr>
<td><strong>InitializeDescription(Object, UriSchemeKeyedCollection)</strong></td>
<td>Initializes a description of the service hosted based on its instance and specified base addresses.</td>
<td>ServiceHost</td>
</tr>
<tr>
<td><strong>InitializeDescription(Type, UriSchemeKeyedCollection)</strong></td>
<td>Initializes a description of the service hosted based on its type and specified base addresses.</td>
<td>ServiceHost</td>
</tr>
<tr>
<td><strong>InitializeRuntime</strong></td>
<td>Initializes the runtime for the service host.</td>
<td>ServiceHostBase</td>
</tr>
<tr>
<td><strong>LoadConfigurationSection</strong></td>
<td>Loads the service element from the configuration.</td>
<td>ServiceHostBase</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
<td></td>
</tr>
<tr>
<td>OnAbort</td>
<td>Aborts the service. (Inherited from <code>ServiceHostBase</code>.)</td>
<td></td>
</tr>
<tr>
<td>OnBeginClose</td>
<td>Begins an asynchronous operation invoked on the close of the service host. (Inherited from <code>ServiceHostBase</code>.)</td>
<td></td>
</tr>
<tr>
<td>OnBeginOpen</td>
<td>Begins an asynchronous operation invoked on the opening of the service host. (Inherited from <code>ServiceHostBase</code>.)</td>
<td></td>
</tr>
<tr>
<td>OnClose</td>
<td>Closes down the hosted service, including their channel dispatchers and associated instance contexts and listeners. (Inherited from <code>ServiceHostBase</code>.)</td>
<td></td>
</tr>
<tr>
<td>OnClosed</td>
<td>Disposes of disposable services that are being hosted when the service host is closed. (Inherited from <code>ServiceHostBase</code>.)</td>
<td></td>
</tr>
<tr>
<td>OnClosing</td>
<td>Invoked during the transition of a communication object into the closing state. (Inherited from <code>CommunicationObject</code>.)</td>
<td></td>
</tr>
<tr>
<td>OnEndClose</td>
<td>Completes an asynchronous operation invoked on the closing of the service host. (Inherited from <code>ServiceHostBase</code>.)</td>
<td></td>
</tr>
<tr>
<td>OnEndOpen</td>
<td>Completes an asynchronous operation invoked on the opening of the service host. (Inherited from <code>ServiceHostBase</code>.)</td>
<td></td>
</tr>
<tr>
<td>Event/Method</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>OnFaulted</strong></td>
<td>Inserts processing on a communication object after transitions to the faulted state due to the invocation of a synchronous operation. (Inherited from <code>CommunicationObject</code>.)</td>
<td></td>
</tr>
<tr>
<td><strong>OnOpen</strong></td>
<td>Opens the channel dispatchers.                                              (Inherited from <code>ServiceHostBase</code>.)</td>
<td></td>
</tr>
<tr>
<td><strong>OnOpened</strong></td>
<td>Gets the service credentials, authentication and authorization behavior for the hosted service. (Inherited from <code>ServiceHostBase</code>.)</td>
<td></td>
</tr>
<tr>
<td><strong>OnOpening</strong></td>
<td>Invoked during the transition of a communication object into the opening state. (Overrides <code>CommunicationObject</code>.)</td>
<td></td>
</tr>
<tr>
<td><strong>Open()</strong></td>
<td>Causes a communication object transition from the created state to the opened state. (Inherited from <code>CommunicationObject</code>.)</td>
<td></td>
</tr>
<tr>
<td><strong>Open(TimeSpan)</strong></td>
<td>Causes a communication object transition from the created state to the opened state within a specified interval of time. (Inherited from <code>CommunicationObject</code>.)</td>
<td></td>
</tr>
<tr>
<td><strong>ReleasePerformanceCounters</strong></td>
<td>(Inherited from <code>ServiceHostBase</code>.)</td>
<td></td>
</tr>
<tr>
<td><strong>SetEndpointAddress</strong></td>
<td>Sets the endpoint address of the specified endpoint to the current address. (Inherited from <code>ServiceHostBase</code>.)</td>
<td></td>
</tr>
<tr>
<td><strong>ThrowIfDisposed</strong></td>
<td>Throws an exception if the communication object is disposed.</td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>ThrowIfDisposedOrImmutable</strong></td>
<td>Throws an exception if communication object property is not set to the state.</td>
<td></td>
</tr>
<tr>
<td><strong>ThrowIfDisposedOrNotOpen</strong></td>
<td>Throws an exception if communication object is not in the <strong>Opened</strong> state.</td>
<td></td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
<td></td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication</td>
<td>Gets the service authentication behavior. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td>Authorization</td>
<td>Gets the authorization behavior for the service hosted. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td>BaseAddresses</td>
<td>Gets the base addresses used by the hosted service. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td>ChannelDispatchers</td>
<td>Gets the collection of channel dispatchers used by the service host. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td>CloseTimeout</td>
<td>Gets or sets the interval of time allowed for the service host to close. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td>Container</td>
<td>Gets or sets the container.</td>
</tr>
<tr>
<td>Credentials</td>
<td>Gets the credential for the service hosted. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td>DefaultCloseTimeout</td>
<td>Gets the default interval of time allowed for the service host to close. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td>DefaultOpenTimeout</td>
<td>Gets the default interval of time allowed for the service host to open. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td>Description</td>
<td>Gets the description of the service hosted. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td>Extensions</td>
<td>Gets the extensions for the current specified service host. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td>ImplementedContracts</td>
<td>Retrieves the contracts implemented by the service hosted.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>IsDisposed</td>
<td>Gets a value that indicates whether the communication object has been disposed.</td>
</tr>
<tr>
<td>ManualFlowControlLimit</td>
<td>Gets or sets the flow control limit for messages received by the service hosted.</td>
</tr>
<tr>
<td>OpenTimeout</td>
<td>Gets or sets the interval of time allowed for the service host to open.</td>
</tr>
<tr>
<td>SingletonInstance</td>
<td>Gets the singleton instance of the hosted service.</td>
</tr>
<tr>
<td>State</td>
<td>Gets a value that indicates the current state of the communication object.</td>
</tr>
<tr>
<td>ThisLock</td>
<td>Gets the mutually exclusive lock that protects the class instance during a state transition.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Closed</strong></td>
<td>Occurs when a communication object transitions into the closed state.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from CommunicationObject.)</td>
</tr>
<tr>
<td><strong>Closing</strong></td>
<td>Occurs when a communication object transitions into the closing state.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from CommunicationObject.)</td>
</tr>
<tr>
<td><strong>Faulted</strong></td>
<td>Occurs when a communication object transitions into the faulted state.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from CommunicationObject.)</td>
</tr>
<tr>
<td><strong>Opened</strong></td>
<td>Occurs when a communication object transitions into the opened state.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from CommunicationObject.)</td>
</tr>
<tr>
<td><strong>Opening</strong></td>
<td>Occurs when a communication object transitions into the opening state.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from CommunicationObject.)</td>
</tr>
<tr>
<td><strong>UnknownMessageReceived</strong></td>
<td>Occurs when an unknown message is received.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from ServiceHostBase.)</td>
</tr>
</tbody>
</table>
See Also

UnityServiceHost Class
Optimization.Unity Namespace
Optimization Framework

**UnityServiceHost Constructor**

[UnityServiceHost Class](#) [See Also](#) [Send Feedback](#)
# Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>UnityServiceHost()</code></td>
<td>Initializes a new instance of the <code>UnityServiceHost</code> class.</td>
</tr>
<tr>
<td><code>UnityServiceHost(Type, Uri[])</code></td>
<td>Initializes a new instance of the <code>UnityServiceHost</code> class.</td>
</tr>
</tbody>
</table>
See Also

UnityServiceHost Class
UnityServiceHost Members
Optimization.Unity Namespace
Initializes a new instance of the UnityServiceHost class.

**Namespace:** Optimization.Unity  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public UnityServiceHost()</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Sub New</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public: UnityServiceHost()</code></td>
</tr>
</tbody>
</table>
See Also

UnityServiceHost Class
UnityServiceHost Overload
Optimization.Unity Namespace
Initializes a new instance of the `UnityServiceHost` class.

**Namespace:** [Optimization.Unity](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

#### C#

```csharp
public UnityServiceHost(
    Type serviceType,
    params Uri[] baseAddresses
)
```

#### Visual Basic

```vbnet
Public Sub New (_
    serviceType As Type, _
    ParamArray baseAddresses As Uri() _
)
```

#### Visual C++

```cpp
public:
UnityServiceHost(
    Type^ serviceType,  
    ... array<Uri^>^ baseAddresses
)
```

### Parameters

- **serviceType**
  - Type: `System.Type`
  - Type of the service.

- **baseAddresses**
  - Type: `System.Uri[]`
  - The base addresses.
See Also

UnityServiceHost Class
UnityServiceHost Overload
Optimization.Unity Namespace
The **UnityServiceHost** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abort</td>
<td>Causes a communication object to transition immediately from its current state into the closing state. (Inherited from CommunicationObject)</td>
</tr>
<tr>
<td>AddBaseAddress</td>
<td>Adds a base address to the service host. (Inherited from ServiceHostBase)</td>
</tr>
<tr>
<td>AddDefaultEndpoints</td>
<td>Adds service endpoints with addresses in each contract in the service host with the default binding. (Inherited from ServiceHostBase)</td>
</tr>
<tr>
<td>AddServiceEndpoint(ServiceEndpoint)</td>
<td>Adds the specified service endpoint to the hosted service. (Inherited from Service)</td>
</tr>
<tr>
<td>AddServiceEndpoint(Type, Binding, String)</td>
<td>Adds a service endpoint to the service with a specified contract, binding, and endpoint address. (Inherited from Service)</td>
</tr>
<tr>
<td>AddServiceEndpoint(Type, Binding, Uri)</td>
<td>Adds a service endpoint to the service with a specified contract, binding, and URI that contains the endpoint address. (Inherited from Service)</td>
</tr>
<tr>
<td>AddServiceEndpoint(String, Binding, String)</td>
<td>Adds a service endpoint to the service with a specified contract, binding, and endpoint address. (Inherited from Service)</td>
</tr>
<tr>
<td>AddServiceEndpoint(String, Binding, Uri)</td>
<td>Adds a service endpoint to the service with a specified contract, binding, and a URI that contains the endpoint address. (Inherited from Service)</td>
</tr>
<tr>
<td>Method Description</td>
<td>Inherited From</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><strong>AddServiceEndpoint</strong>&lt;br&gt;(Type, Binding, String, Uri)</td>
<td>ServiceHostBase</td>
</tr>
<tr>
<td>Adds a service endpoint to the hosted service with a specified contract, binding, an endpoint address, and a URI on which the service listens.</td>
<td></td>
</tr>
<tr>
<td><strong>AddServiceEndpoint</strong>&lt;br&gt;(Type, Binding, Uri, Uri)</td>
<td>ServiceHost</td>
</tr>
<tr>
<td>Adds a service endpoint to the hosted service with a specified contract, binding, a URI that contains the endpoint address, and a URI on which the service listens.</td>
<td></td>
</tr>
<tr>
<td><strong>AddServiceEndpoint</strong>&lt;br&gt;(String, Binding, String, Uri)</td>
<td>ServiceHostBase</td>
</tr>
<tr>
<td>Adds a service endpoint to the hosted service with a specified contract, binding, endpoint address, and a URI that contains the address at which it listens.</td>
<td></td>
</tr>
<tr>
<td><strong>AddServiceEndpoint</strong>&lt;br&gt;(String, Binding, Uri, Uri)</td>
<td>ServiceHostBase</td>
</tr>
<tr>
<td>Adds a service endpoint to the hosted service with the specified contract, binding, and URIs that contain the endpoint and listening addresses.</td>
<td></td>
</tr>
<tr>
<td><strong>ApplyConfiguration</strong></td>
<td>ServiceHostBase</td>
</tr>
<tr>
<td>Loads the service description information from the configuration file and applies it to the service being constructed.</td>
<td></td>
</tr>
<tr>
<td><strong>BeginClose</strong>&lt;br&gt;(AsyncCallback, Object)</td>
<td>CommunicationObject</td>
</tr>
<tr>
<td>Begins an asynchronous operation to close a communication object.</td>
<td></td>
</tr>
<tr>
<td><strong>BeginClose</strong>&lt;br&gt;(TimeSpan, AsyncCallback, Object)</td>
<td>CommunicationObject</td>
</tr>
<tr>
<td>Begins an asynchronous operation to close a communication object with a specified timeout.</td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>BeginOpen(AsyncCallback, Object)</strong></td>
<td>Begins an asynchronous operation to open a communication object. (Inherited from CommunicationObject)</td>
</tr>
<tr>
<td><strong>BeginOpen(TimeSpan, AsyncCallback, Object)</strong></td>
<td>Begins an asynchronous operation to open a communication object within a specified interval of time. (Inherited from CommunicationObject)</td>
</tr>
<tr>
<td><strong>Close()</strong></td>
<td>Causes a communication object to transition from its current state into the closed state. (Inherited from CommunicationObject)</td>
</tr>
<tr>
<td><strong>Close(TimeSpan)</strong></td>
<td>Causes a communication object to transition from its current state into the closed state within a specified interval of time. (Inherited from CommunicationObject)</td>
</tr>
<tr>
<td><strong>CreateDescription</strong></td>
<td>Creates a description of the service hosted. (Inherited from Service)</td>
</tr>
<tr>
<td><strong>EndClose</strong></td>
<td>Completes an asynchronous operation to close a communication object. (Inherited from CommunicationObject)</td>
</tr>
<tr>
<td><strong>EndOpen</strong></td>
<td>Completes an asynchronous operation to open a communication object. (Inherited from CommunicationObject)</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified Object is equal to the current Object. (Inherited from Object)</td>
</tr>
<tr>
<td><strong>Fault</strong></td>
<td>Causes a communication object to transition from its current state into the faulted state.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an Object to attempt to free resources and perform cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetCommunicationObjectType</strong></td>
<td>Gets the type of communication object. (Inherited from CommunicationObject.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>IncrementManualFlowControlLimit</strong></td>
<td>Increases the limit on the flow rate of messages to the hosted service by a specified increment. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>InitializeDescription(UriSchemeKeyedCollection)</strong></td>
<td>Creates and initializes the service host with the contract and service descriptions. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td><strong>InitializeDescription(Object, UriSchemeKeyedCollection)</strong></td>
<td>Initializes a description of the service hosted based on its instance and specified base addresses. (Inherited from ServiceHost.)</td>
</tr>
<tr>
<td><strong>InitializeDescription(Type, UriSchemeKeyedCollection)</strong></td>
<td>Initializes a description of the service hosted based on its type and specified base addresses. (Inherited from ServiceHost.)</td>
</tr>
<tr>
<td><strong>InitializeRuntime</strong></td>
<td>Initializes the runtime for the service host. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td><strong>LoadConfigurationSection</strong></td>
<td>Loads the service element from the configuration file. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current object. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>OnAbort</strong></td>
<td>Aborts the service. (Inherited from <strong>Service</strong>.)</td>
</tr>
<tr>
<td><strong>OnBeginClose</strong></td>
<td>Begins an asynchronous operation invoked on the close of the service host. (Inherited from <strong>ServiceHostBase</strong>.)</td>
</tr>
<tr>
<td><strong>OnBeginOpen</strong></td>
<td>Begins an asynchronous operation invoked on the opening of the service host. (Inherited from <strong>ServiceHostBase</strong>.)</td>
</tr>
<tr>
<td><strong>OnClose</strong></td>
<td>Closes down the hosted service, including their channel dispatchers and associated instance contexts and listeners. (Inherited from <strong>Service</strong>.)</td>
</tr>
<tr>
<td><strong>OnClosed</strong></td>
<td>Disposes of disposable services that are being hosted when the service host is closed. (Inherited from <strong>Service</strong>.)</td>
</tr>
<tr>
<td><strong>OnClosing</strong></td>
<td>Invoked during the transition to the closing state of a communication object. (Inherited from <strong>CommunicationObject</strong>.)</td>
</tr>
<tr>
<td><strong>OnEndClose</strong></td>
<td>Completes an asynchronous operation invoked on the closing of the service host. (Inherited from <strong>Service</strong>.)</td>
</tr>
<tr>
<td><strong>OnEndOpen</strong></td>
<td>Completes an asynchronous operation invoked on the opening of the service host. (Inherited from <strong>Service</strong>.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>OnFaulted</td>
<td>Inserts processing on a communication object after it transitions to the faulted state due to the invocation of a synchronous operation. (Inherited from CommunicationObject)</td>
</tr>
<tr>
<td>OnOpen</td>
<td>Opens the channel dispatchers. (Inherited from ServiceHostBase)</td>
</tr>
<tr>
<td>OnOpened</td>
<td>Gets the service credentials authentication and authorization behavior for the hosted service. (Inherited from ServiceHostBase)</td>
</tr>
<tr>
<td>OnOpening</td>
<td>Invoked during the transition of a communication object into the opening state. (Overrides CommunicationObject)</td>
</tr>
<tr>
<td>Open()</td>
<td>Causes a communication object transition from the created state into the opened state. (Inherited from CommunicationObject)</td>
</tr>
<tr>
<td>Open(TimeSpan)</td>
<td>Causes a communication object transition from the created state into the opened state within a specified interval of time. (Inherited from CommunicationObject)</td>
</tr>
<tr>
<td>ReleasePerformanceCounters</td>
<td>(Inherited from ServiceHostBase)</td>
</tr>
<tr>
<td>SetEndpointAddress</td>
<td>Sets the endpoint address of the specified endpoint to the specified address. (Inherited from ServiceHostBase)</td>
</tr>
<tr>
<td>ThrowIfDisposed</td>
<td>Throws an exception if the communication object is disposed. (Inherited from ServiceHostBase)</td>
</tr>
<tr>
<td><strong>ThrowIfDisposedOrImmutable</strong></td>
<td>Throws an exception if communication object property is not set to the state. (Inherited from CommunicationObject)</td>
</tr>
<tr>
<td><strong>ThrowIfDisposedOrNotOpen</strong></td>
<td>Throws an exception if communication object state. (Inherited from CommunicationObject)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a String that represents the current Object. (Inherited from Object)</td>
</tr>
</tbody>
</table>
See Also

UnityServiceHost Class
Optimization.Unity Namespace
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddServiceEndpoint(ServiceEndpoint)</td>
<td>Adds the specified service endpoint to the hosted service. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td>AddServiceEndpoint(Type, Binding, String)</td>
<td>Adds a service endpoint to the hosted service with a specified contract, binding, and endpoint address. (Inherited from ServiceHost.)</td>
</tr>
<tr>
<td>AddServiceEndpoint(Type, Binding, Uri)</td>
<td>Adds a service endpoint to the hosted service with a specified contract, binding, and URI that contains the endpoint address. (Inherited from ServiceHost.)</td>
</tr>
<tr>
<td>AddServiceEndpoint(String, Binding, String)</td>
<td>Adds a service endpoint to the hosted service with a specified contract, binding, and endpoint address. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td>AddServiceEndpoint(String, Binding, Uri)</td>
<td>Adds a service endpoint to the hosted service with a specified contract, binding, and a URI that contains the endpoint address. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td>AddServiceEndpoint(Type, Binding, String, Uri)</td>
<td>Adds a service endpoint to the hosted service with a specified contract, binding, and endpoint address. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>AddServiceEndpoint</strong>&lt;br&gt;(Type, Binding, Uri, Uri)</td>
<td>Adds a service endpoint to the hosted service with a specified contract, binding, a URI that contains the endpoint address, and a URI on which the service listens. (Inherited from ServiceHost.)</td>
</tr>
<tr>
<td><strong>AddServiceEndpoint</strong>&lt;br&gt;(String, Binding, String, Uri)</td>
<td>Adds a service endpoint to the hosted service with a specified contract, binding, endpoint address and URI that contains the address at which it listens. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td><strong>AddServiceEndpoint</strong>&lt;br&gt;(String, Binding, Uri, Uri)</td>
<td>Adds a service endpoint to the hosted service with the specified contract, binding, and URIs that contain the endpoint and listening addresses. (Inherited from ServiceHostBase.)</td>
</tr>
</tbody>
</table>

An endpoint address, and a URI on which the service listens. (Inherited from ServiceHost.)
See Also

UnityServiceHost Class
UnityServiceHost Members
Optimization.Unity Namespace
Optimization Framework

UnityServiceHost.BeginClose Method

UnityServiceHost Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>BeginClose(AsyncCallback, Object)</code></td>
<td>Begins an asynchronous operation to close a communication object. (Inherited from <code>CommunicationObject</code>.)</td>
</tr>
<tr>
<td><code>BeginClose(TimeSpan, AsyncCallback, Object)</code></td>
<td>Begins an asynchronous operation to close a communication object with a specified timeout. (Inherited from <code>CommunicationObject</code>.)</td>
</tr>
</tbody>
</table>
See Also

UnityServiceHost Class
UnityServiceHost Members
Optimization.Unity Namespace
Optimization Framework

UnityServiceHost.BeginOpen Method

UnityServiceHost Class See Also Send Feedback
<table>
<thead>
<tr>
<th><strong>Name</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>BeginOpen(AsyncCallback, Object)</code></td>
<td>Begins an asynchronous operation to open a communication object. (Inherited from <code>CommunicationObject</code>.)</td>
</tr>
<tr>
<td><code>BeginOpen(TimeSpan, AsyncCallback, Object)</code></td>
<td>Begins an asynchronous operation to open a communication object within a specified interval of time. (Inherited from <code>CommunicationObject</code>.)</td>
</tr>
</tbody>
</table>
See Also

UnityServiceHost Class
UnityServiceHost Members
Optimization.Unity Namespace
Optimization Framework

UnityServiceHost.Close Method

UnityServiceHost Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Close()</td>
<td>Causes a communication object to transition from its current state into the</td>
</tr>
<tr>
<td></td>
<td>closed state.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">CommunicationObject</a>.)</td>
</tr>
<tr>
<td>Close(TimeSpan)</td>
<td>Causes a communication object to transition from its current state into the</td>
</tr>
<tr>
<td></td>
<td>closed state within a specified interval of time.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">CommunicationObject</a>.)</td>
</tr>
</tbody>
</table>
See Also

UnityServiceHost Class
UnityServiceHost Members
Optimization.Unity Namespace
Optimization Framework

**UnityServiceHost.InitializeDescription Method**

[UnityServiceHost Class] [See Also] [Send Feedback]
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InitializeDescription(UriSchemeKeyedCollection)</td>
<td>Creates and initializes the service host with the contract and service descriptions. (Inherited from ServiceHostBase.)</td>
</tr>
<tr>
<td>InitializeDescription(Object, UriSchemeKeyedCollection)</td>
<td>Initializes a description of the service hosted based on its instance and specified base addresses. (Inherited from ServiceHost.)</td>
</tr>
<tr>
<td>InitializeDescription(Type, UriSchemeKeyedCollection)</td>
<td>Initializes a description of the service hosted based on its type and specified base addresses. (Inherited from ServiceHost.)</td>
</tr>
</tbody>
</table>
See Also

UnityServiceHost Class
UnityServiceHost Members
Optimization.Unity Namespace
Invoked during the transition of a communication object into the opening state.

**Namespace**: Optimization.Unity

**Assembly**: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td>protected override void OnOpening()</td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td>Protected Overrides Sub OnOpening</td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td>protected: virtual void OnOpening() override</td>
</tr>
</tbody>
</table>
See Also

UnityServiceHost Class
Optimization.Unity Namespace
Optimization Framework

UnityServiceHost.Open Method

UnityServiceHost Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>![bullseye] Open()</td>
<td>Causes a communication object to transition from the created state into the opened state. (Inherited from CommunicationObject.)</td>
</tr>
<tr>
<td>![bullseye] Open(TimeSpan)</td>
<td>Causes a communication object to transition from the created state into the opened state within a specified interval of time. (Inherited from CommunicationObject.)</td>
</tr>
</tbody>
</table>
See Also

UnityServiceHost Class
UnityServiceHost Members
Optimization.Unity Namespace
Optimization Framework

**UnityServiceHost Properties**

See Also Send Feedback

The **UnityServiceHost** type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication</td>
<td>Gets the service authentication behavior. (Inherited from <code>ServiceHostBase</code>.)</td>
</tr>
<tr>
<td>Authorization</td>
<td>Gets the authorization behavior for the service hosted. (Inherited from <code>ServiceHostBase</code>.)</td>
</tr>
<tr>
<td>BaseAddresses</td>
<td>Gets the base addresses used by the hosted service. (Inherited from <code>ServiceHostBase</code>.)</td>
</tr>
<tr>
<td>ChannelDispatchers</td>
<td>Gets the collection of channel dispatchers used by the service host. (Inherited from <code>ServiceHostBase</code>.)</td>
</tr>
<tr>
<td>CloseTimeout</td>
<td>Gets or sets the interval of time allowed for the service host to close. (Inherited from <code>ServiceHostBase</code>.)</td>
</tr>
<tr>
<td>Container</td>
<td>Gets or sets the container.</td>
</tr>
<tr>
<td>Credentials</td>
<td>Gets the credential for the service hosted. (Inherited from <code>ServiceHostBase</code>.)</td>
</tr>
<tr>
<td>DefaultCloseTimeout</td>
<td>Gets the default interval of time allowed for the service host to close. (Inherited from <code>ServiceHostBase</code>.)</td>
</tr>
<tr>
<td>DefaultOpenTimeout</td>
<td>Gets the default interval of time allowed for the service host to open. (Inherited from <code>ServiceHostBase</code>.)</td>
</tr>
<tr>
<td>Description</td>
<td>Gets the description of the service hosted. (Inherited from <code>ServiceHostBase</code>.)</td>
</tr>
<tr>
<td>Extensions</td>
<td>Gets the extensions for the current specified service host. (Inherited from <code>ServiceHostBase</code>.)</td>
</tr>
<tr>
<td>ImplementedContracts</td>
<td>Retrieves the contracts implemented by the service hosted.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>IsDisposed</td>
<td>Gets a value that indicates whether the communication object has been disposed.</td>
</tr>
<tr>
<td>ManualFlowControlLimit</td>
<td>Gets or sets the flow control limit for messages received by the service hosted.</td>
</tr>
<tr>
<td>OpenTimeout</td>
<td>Gets or sets the interval of time allowed for the service host to open.</td>
</tr>
<tr>
<td>SingletonInstance</td>
<td>Gets the singleton instance of the hosted service.</td>
</tr>
<tr>
<td>State</td>
<td>Gets a value that indicates the current state of the communication object.</td>
</tr>
<tr>
<td>ThisLock</td>
<td>Gets the mutually exclusive lock that protects the class instance during a state transition.</td>
</tr>
</tbody>
</table>
See Also

UnityServiceHost Class
Optimization.Unity Namespace
Optimization Framework

UnityServiceHost.Container Property

Gets or sets the container.

**Namespace:** [Optimization.Unity](#)

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
</table>
| **C#**   | ```
public IUnityContainer Container { get; set; }
``` |
| **Visual Basic** | ```
Public Property Container As IUnityContainer
    Get
    Set
``` |
| **Visual C++** | ```
public:
    property IUnityContainer^ Container {
        IUnityContainer^ get ()
        void set (IUnityContainer^ value);
    }
``` |

### Field Value

The container.
See Also

UnityServiceHost Class
Optimization.Unity Namespace
The **UnityServiceHost** type exposes the following members.
### Events

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Closed</td>
<td>Occurs when a communication object transitions into the closed state. (Inherited from CommunicationObject.)</td>
</tr>
<tr>
<td>Closing</td>
<td>Occurs when a communication object transitions into the closing state. (Inherited from CommunicationObject.)</td>
</tr>
<tr>
<td>Faulted</td>
<td>Occurs when a communication object transitions into the faulted state. (Inherited from CommunicationObject.)</td>
</tr>
<tr>
<td>Opened</td>
<td>Occurs when a communication object transitions into the opened state. (Inherited from CommunicationObject.)</td>
</tr>
<tr>
<td>Opening</td>
<td>Occurs when a communication object transitions into the opening state. (Inherited from CommunicationObject.)</td>
</tr>
<tr>
<td>UnknownMessageReceived</td>
<td>Occurs when an unknown message is received. (Inherited from ServiceHostBase.)</td>
</tr>
</tbody>
</table>
See Also

UnityServiceHost Class
Optimization.Unity Namespace
Optimization Framework

UnityServiceHostFactory Class

Allows to create unity service hosts.

Namespace: Optimization.Unity
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C#</td>
<td><code>public abstract class UnityServiceHostFactory : ServiceHostFactory</code></td>
</tr>
<tr>
<td>Visual Basic</td>
<td><code>Public MustInherit Class UnityServiceHostFactory _ Inherits ServiceHostFactoryBase</code></td>
</tr>
<tr>
<td>Visual C++</td>
<td><code>public ref class UnityServiceHostFactory abstract</code></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

- System.Object
- System.ServiceModel.Activation.ServiceHostFactoryBase
- Optimization.Unity.UnityServiceHostFactory
- Optimization.Unity.DefaultHostFactory
See Also

UnityServiceHostFactory Members
Optimization.Unity Namespace
The **UnityServiceHostFactory** type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UnityServiceHostFactory</td>
<td>Initializes a new instance of the UnityServiceHostFactory class.</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateServiceHost</td>
<td>When overridden in a derived class, creates a ServiceHostBase with a specific base address using custom initiation data. (Overrides ServiceHostFactoryBase.CreateServiceHost(String, Uri[]).)</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified Object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a String that represents the current Object. (Inherited from Object.)</td>
</tr>
</tbody>
</table>
See Also

UnityServiceHostFactory Class
Optimization.Unity Namespace
Optimization Framework

UnityServiceHostFactory Constructor

Namespace: Optimization.Unity
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0

Initializes a new instance of the UnityServiceHostFactory class.
## Syntax

### C#

```
protected UnityServiceHostFactory(
    IUnityConfiguration unityConfiguration
)
```

### Visual Basic

```
Protected Sub New ( _
    unityConfiguration As IUnityConfiguration _
)
```

### Visual C++

```
protected:
UnityServiceHostFactory(
    IUnityConfiguration^ unityConfiguration
)
```

## Parameters

- **unityConfiguration**
  - Type: `Optimization.Unity.IUnityConfiguration`
  - The unity configuration.
See Also

UnityServiceHostFactory Class
Optimization.Unity Namespace
The **UnityServiceHostFactory** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateServiceHost</td>
<td>When overridden in a derived class, creates a ServiceHostBase with a specific base address using custom initiation data. (Overrides ServiceHostFactoryBase.CreateServiceHost(String, Uri[]).)</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified Object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a String that represents the current Object. (Inherited from Object.)</td>
</tr>
</tbody>
</table>
See Also

UnityServiceHostFactory Class
Optimization.Unity Namespace
When overridden in a derived class, creates a `IServiceHostBase` with a specific base address using custom initiation data.

**Namespace:** Optimization.Unity  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
### Syntax

**C#**

```csharp
public override ServiceHostBase CreateServiceHost(
    string constructorString,
    Uri[] baseAddresses
)
```

**Visual Basic**

```vbnet
Public Overrides Function CreateServiceHost ( _
    constructorString As String, _
    baseAddresses As Uri() _
) As ServiceHostBase
```

**Visual C++**

```cpp
public:
    virtual ServiceHostBase^ CreateServiceHost(
        String^ constructorString,
        array<Uri^>^ baseAddresses
    ) override
```

### Parameters

**constructorString**
Type: `System.String`

The initialization data that is passed to the `ServiceHostBase` instance being constructed by the factory.

**baseAddresses**
Type: `System.Uri[]`

An `Array` of type `Uri` that contains the base addresses of the host.
Return Value

The `ServiceHostBase` object with the specified base addresses and initialized with the custom initiation data.
See Also

UnityServiceHostFactory Class
Optimization.Unity Namespace
Optimization Framework

WcfUnityConfiguration Class

Allows configuration of a unity container in a WCF context

Namespace: Optimization.Unity
Assembly: Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
</tr>
<tr>
<td><code>public class WcfUnityConfiguration : IUnityConfiguration</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
</tr>
<tr>
<td><code>Public Class WcfUnityConfiguration _ Implements IUnityConfiguration</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
</tr>
<tr>
<td><code>public ref class WcfUnityConfiguration : IUnityConfiguration</code></td>
</tr>
</tbody>
</table>
Inheritance Hierarchy

System.Object
Optimization.Unity.WcfUnityConfiguration
See Also

WcfUnityConfiguration Members
Optimization.Unity Namespace
The `WcfUnityConfiguration` type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WcfUnityConfiguration</td>
<td>Initializes a new instance of the WcfUnityConfiguration class</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Configure</code></td>
<td>Configures the specified container.</td>
</tr>
<tr>
<td><code>Equals</code></td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><code>Finalize</code></td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><code>GetHashCode</code></td>
<td>Serves as a hash function for a particular type. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><code>GetType</code></td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><code>MemberwiseClone</code></td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><code>ToString</code></td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
</tbody>
</table>
See Also

WcfUnityConfiguration Class
Optimization.Unity Namespace
Initializes a new instance of the `WcfUnityConfiguration` class

**Namespace:** Optimization.Unity  
**Assembly:** Optimization.Framework (in Optimization.Framework.dll)  
**Version:** 0.3.1036.0
### Syntax

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C#</strong></td>
<td><code>public WcfUnityConfiguration()</code></td>
</tr>
<tr>
<td><strong>Visual Basic</strong></td>
<td><code>Public Sub New</code></td>
</tr>
<tr>
<td><strong>Visual C++</strong></td>
<td><code>public: WcfUnityConfiguration()</code></td>
</tr>
</tbody>
</table>
See Also

WcfUnityConfiguration Class
Optimization.Unity Namespace
The **WcfUnityConfiguration** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Configure</strong></td>
<td>Configures the specified container.</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified Object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a String that represents the current Object. (Inherited from Object.)</td>
</tr>
</tbody>
</table>
See Also

WcfUnityConfiguration Class
Optimization.Unity Namespace
WcfUnityConfiguration.Configure Method

Configures the specified container.

**Namespace:** Optimization.Unity

**Assembly:** Optimization.Framework (in Optimization.Framework.dll) Version: 0.3.1036.0
## Syntax

### C#

```csharp
public void Configure(
    IUnityContainer container
)
```

### Visual Basic

```vbnet
Public Sub Configure (_
    container As IUnityContainer _
)
```

### Visual C++

```cpp
public:
virtual void Configure(
    IUnityContainer^ container
) sealed
```

## Parameters

- **container**
  - Type: **IUnityContainer**
  - The container.

## Implements

- **IUnityConfiguration.Configure(IUnityContainer)**
See Also

[Link: WcfUnityConfiguration Class]
[Link: Optimization.Unity Namespace]