Xface Core Library File List

Here is a list of all documented files with brief descriptions:

ApplicationBase.h	*Base for application dev classes*
BmpFile.h	*BMP file loader for textures*
DeformableGeometry.h	*Deformation enabled geometry class*
Drawable.h	*Stores the information for drawable objects*
Entity.h	*Represents an entity visible in the scene, includes drawables, ai, etc.*
FaceBase.h	*Main face object*
FapFile.h	*MPEG-4 FAPs loader from disk*
FDP.h	*FDP information is here*
FDPFile.h	*FDP file streaming*
FDPItem.h	*FDP item data representation*
FDPLoader.h	*FDP Loader*
FDPWriter.h	*FDP Writer*
Geometry.h	*Defines interface for the mesh geometry*
IFapStream.h	*Interface for MPEG-4 FAPs streaming*
IInfluenceCalculator.h	*Interface for deforming indexed face set meshes*
IModelLoader.h	*Base class for the scene graph loaders*
IndexedFaceSet.h	*Indexed mesh geometry implemented*
InfluenceCalculatorMaker.h	*InfluenceCalculator pluggable factory maker*
IRenderer.h	*Interface for renderer*
ISound.h	*Sound player interface*
ITexture.h	*Base class for texture mapping*
ITextureFile.h	*Interface for image file loaders for textures*
ITextureLoader.h	*Base class for texture loading to any API*
ITimedController.h	*ABS for timed controllers*
ITimer.h	*Timer interface*
KeyframeInterpolator.h	*Keyframe interpolation done here*
Matrix4.h	*4x4 Matrix class*
MeshManager.h	*Meshes are stored and managed here*
ModelCamera.h	*Camera class*
ModelFileFactory.h	*3d Models are loaded through here*
MorphController.h	*Morph targets controller*

MorphTarget.h	*Morph target definition*
NamedObj.h	*Utility class for named classes*
Notification.h	*Notification message to server for remote communication*
OBJLoader.h	*Wavefront OBJ loader*
RaisedCosInfluence.h	*Raised Cosine deformation*
Ray3.h	*Ray representation in 3D*
Rectangle.h	*Rectangle representation*
RendererGL.h	*OpenGL renderer*
RenderList.h	*Render list storing Drawables and calling proper IRenderer methods*
Task.h	*Task related*
TaskHandlerBase.h	*Task handler base*
Texture2D.h	*2D texture mapping*
TextureLoaderGL.h	*Loads the textures into OpenGL*
TextureManager.h	*Textures are stored and managed here*
TgaFile.h	*TGA file loader for textures*
Transform.h	*Transform node for scenegraph*
Vector3.h	*3D Vector class*
VertexStruct.h	*Vertex structures are here*
VRML1Loader.h	*VRML1 Loader*
VRML97Loader.h	*Loads VRML97 files*
XercesString.h	*Xerces string operations*
XLineSegment.h	*Representation of a 3D line segment*
XMath.h	*Intersection related functions*
XMLUtils.h	*XML related functions*
XTriangle.h	*Triangle representation*

ApplicationBase.h File Reference

Detailed Description

```
#include <list> #include <sstream>
#include <boost/shared_ptr.hpp>
#include <XEngine/RenderManager.h>
#include <XFaceApp/TaskHandlerBase.h>
#include <XFaceApp/Notification.h>
#include <XFace/FaceBase.h>
#include <XFaceApp/ISound.h>
#include <XEngine/ITimer.h>
#include <XFace/IFapStream.h>
#include <IScriptProcessor.h>
```

Namespaces

namespace	**XFaceApp**

Classes

class	**XFaceApp::ApplicationBase**
	Base for application dev classes. *More...*

BmpFile.h File Reference

Detailed Description

```
#include "ITextureFile.h"
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::BmpFile**
	BMP file loader for textures. More...

DeformableGeometry.h File Reference

Detailed Description

```
#include "IndexedFaceSet.h" #include <cassert>
#include <map>
#include <set>
#include <vector>
#include <algorithm>
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::DeformableGeometry**
	Deformation enabled geometry class. More...

Drawable.h File Reference

Detailed Description

```
#include <string> #include <vector>
#include <cassert>
#include <iostream>
#include "Transform.h"
#include "DeformableGeometry.h"
#include "MeshManager.h"
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::Drawable**
	Stores the information for drawable objects. More...

Entity.h File Reference

Detailed Description

```
#include <list> #include <string>
#include "Drawable.h"
#include "Transform.h"
#include <boost/shared_ptr.hpp>
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::Entity**
	Represents an entity visible in the scene, includes drawables, ai, etc.. More...

Generated on Mon Aug 28 15:39:25 2006 for Xface Core Library by  *1.4.6-NO*

FaceBase.h File Reference

Detailed Description

```
#include <string> #include <map>
#include <vector>
#include <list>
#include <sstream>
#include <boost/shared_ptr.hpp>
#include <XEngine/Entity.h>
#include <XEngine/MorphTarget.h>
#include <XFace/FaceEntity.h>
#include <XFace/FDP.h>
#include <XEngine/ModelFileFactory.h>
#include <XEngine/MeshManager.h>
#include <XEngine/MorphController.h>
#include <XFace/AnimProcessor.h>
```

Namespaces

namespace	**XFace**

Classes

class	**XFace::FaceBase**
	*Main **XFace** class. More...*

FapFile.h File Reference

Detailed Description

```
#include "IFapStream.h" #include <string>
#include <list>
```

Namespaces

namespace	**XFace**

Classes

class	**XFace::FAPFile**
	MPEG-4 FAPs loader from disk. More...

FDP.h File Reference

Detailed Description

```
#include "FDPItem.h" #include <XMath/Vector3.h>
#include <XMath/AxisAngle.h>
#include <list>
#include <string>
```

Namespaces

namespace	**XFace**

Classes

class	**XFace::FAPU** *MPEG4 **FAPU** info is in here. More...*
class	**XFace::FDP** *MPEG4 **FDP** data as a whole. More...*

Generated on Mon Aug 28 15:39:25 2006 for Xface Core Library by  *1.4.6-NO*

FDPFile.h File Reference

Detailed Description

```
#include <tinyxml.h> #include <iostream>
#include <fstream>
#include <string>
#include <list>
#include <boost/shared_ptr.hpp>
#include "FDP.h"
#include <XEngine/ModelFileFactory.h>
#include <XEngine/Entity.h>
```

Namespaces

namespace	**XFace**

Classes

class	**XFace::FDPFile**
	*MPEG-4 **FDP** loader/writer from/to disk. More...*

FDPItem.h File Reference

Detailed Description

```
#include <XFace/IInfluenceCalculator.h> #include <set>
#include <map>
#include <vector>
#include <string>
```

Namespaces

namespace	**XFace**

Classes

class	**XFace::FDPItem**
	*MPEG4 **FDP** data. More...*

FDPLoader.h File Reference

Detailed Description

```
#include <xercesc/dom/DOM.hpp> #include
<xercesc/parsers/XercesDOMParser.hpp>
#include <xercesc/util/XercesDefs.hpp>
#include <boost/shared_ptr.hpp>
#include <iostream>
#include <fstream>
#include "FDP.h"
#include <XEngine/ModelFileFactory.h>
#include <XEngine/Entity.h>
#include <map>
#include <list>
```

Namespaces

namespace	**XFace**

Classes

class	**XFace::FDPLoader** ***FDP*** *Loader.* *More...*

FDPWriter.h File Reference

Detailed Description

```
#include <string> #include <boost/shared_ptr.hpp>
#include <xercesc/dom/DOM.hpp>
#include <XFace/FDP.h>
#include <XEngine/Entity.h>
```

Namespaces

namespace	**XFace**

Geometry.h File Reference

Detailed Description

```
#include "VertexStruct.h" #include <vector>
#include <XMath/Vector3.h>
#include "NamedObj.h"
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::Geometry**
	Defines interface for the mesh geometry. More...

IFapStream.h File Reference

Detailed Description

```
#include <cassert> #include <iostream>
#include "FDP.h"
```

Namespaces

namespace	**XFace**

Classes

class	**XFace::IFapStream**
	Interface for MPEG-4 FAPs streaming. *More...*

IInfluenceCalculator.h File Reference

Detailed Description

#include <XMath/Vector3.h> #include <vector>

Namespaces

namespace	**XFace**

Classes

class	**XFace::IInfluenceCalculator**
	Interface for defining the weights in the zone of influence for deformation. More...

IModelLoader.h File Reference

Detailed Description

```
#include <list> #include <string>
#include <boost/shared_ptr.hpp>
#include "Drawable.h"
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::IModelLoader**
	Base class for the scene graph loaders. More...

IndexedFaceSet.h File Reference

Detailed Description

```
#include "Geometry.h" #include <vector>
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::IndexedFaceSet**
	Indexed mesh geometry implemented. More...

InfluenceCalculatorMaker.h File Reference

Detailed Description

```
#include "IInfluenceCalculator.h" #include <string>
#include <map>
#include <vector>
```

Namespaces

namespace	**XFace**

Classes

class	**XFace::InfluenceCalculatorMaker**
	InfluenceCalculator pluggable factory maker. More...

IRenderer.h File Reference

Detailed Description

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::IRenderer**
	Interface for renderer. More...

ISound.h File Reference

Detailed Description

```
#include <string>
```

Namespaces

namespace	**XFaceApp**

Classes

class	**XFaceApp::ISound**
	Sound player interface. More...

ITexture.h File Reference

Detailed Description

```
#include "NamedObj.h"
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::ITexture**
	Base class for texture mapping. More...

ITextureFile.h File Reference

Detailed Description

```
#include <string>
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::ITextureFile**
	Interface for image file loaders for textures. More...

ITextureLoader.h File Reference

Detailed Description

```
#include <string> #include "ITexture.h"
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::ITextureLoader**
	Base class for texture loading. More...

ITimedController.h File Reference

Detailed Description

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::ITimedController**
	ABS for timed controllers. More...

ITimer.h File Reference

Detailed Description

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::ITimer**
	Timer interface. More...

KeyframeInterpolator.h File Reference

Detailed Description

```
#include <boost/shared_ptr.hpp> #include <algorithm>
#include "Entity.h"
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::KeyframeInterpolator**
	Keyframe interpolator. More...

Matrix4.h File Reference

Detailed Description

```
#include "Vector3.h"
```

Namespaces

namespace	**XMath**

Classes

class	**XMath::Matrix4**
	4x4 Matrix class for float *More...*

MeshManager.h File Reference

Detailed Description

```
#include <map> #include <string>
#include "DeformableGeometry.h"
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::MeshManager**
	Meshes are stored and managed here. More...

ModelCamera.h File Reference

Detailed Description

`#include <XMath/Matrix4.h> #include <XMath/AxisAngle.h>`

Namespaces

namespace	**XFaceApp**

Classes

class	**XFaceApp::ModelCamera** *Camera class. More...*

ModelFileFactory.h File Reference

Detailed Description

```
#include <list> #include <boost/shared_ptr.hpp>
#include "Drawable.h"
#include "BinaryModelBatchLoader.h"
```

Namespaces

namespace	**XEngine**

Classes

struct	**XEngine::MeshInfo** *3D Meshe info More...*
class	**XEngine::ModelFileFactory** *3D Meshes are loaded from here. More...*

MorphController.h File Reference

Detailed Description

```
#include "ITimedController.h" #include "KeyframeInterpolator.h"
#include "MorphTarget.h"
#include <XMath/Perlin.h>
#include <XFace/FDP.h>
#include <string>
#include <list>
#include <algorithm>
#include "MorphChannel.h"
#include "MorphBlender.h"
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::MorphController**
	Morph targets controller. *More...*

MorphTarget.h File Reference

Detailed Description

```
#include "Entity.h" #include <string>
```

Namespaces

namespace	**XEngine**

NamedObj.h File Reference

Detailed Description

```
#include <string>
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::NamedObj**
	Named classes base. More...

Notification.h File Reference

Detailed Description

```
#include <XFaceApp/Task.h> #include <string>
```

Namespaces

namespace	**XFaceApp**

Classes

class	**XFaceApp::Notification**
	Notification *message to server for remote communication.* *More...*

OBJLoader.h File Reference

Detailed Description

```
#include "IModelLoader.h" #include "VertexStruct.h"
#include "DeformableGeometry.h"
#include <vector>
#include <queue>
#include <istream>
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::OBJLoader**
	Loads Wavefront OBJ files for the scene graph. More...

RaisedCosInfluence.h File Reference

Detailed Description

```
#include "IInfluenceCalculator.h" #include
"InfluenceCalculatorMaker.h"
```

Namespaces

namespace	**XFace**

Classes

class	**XFace::RaisedCosInfluence** *Raised cosine deformation base. More...*
class	**XFace::RaisedCosInfluenceSph** *Raised cosine deformation as a sphere. More...*
class	**XFace::RaisedCosInfluenceSphMaker** *Raised cosine influence maker. More...*
class	**XFace::RaisedCosInfluenceWaveX** *Raised cosine deformation as a wave in X direction. More...*
class	**XFace::RaisedCosInfluenceWaveXMaker** *Raised cosine influence maker. More...*
class	**XFace::RaisedCosInfluenceWaveY** *Raised cosine deformation as a wave in Y direction. More...*
class	**XFace::RaisedCosInfluenceWaveYMaker** *Raised cosine Influence maker. More...*
class	**XFace::RaisedCosInfluenceWaveZ** *Raised cosine deformation as a wave in Z direction. More...*
class	**XFace::RaisedCosInfluenceWaveZMaker** *Raised cosine Influence maker. More...*

Ray3.h File Reference

Detailed Description

```
#include "Vector3.h"
```

Namespaces

namespace	**XMath**

Classes

class	**XMath::Ray3**
	3D ray representation More...

Rectangle.h File Reference

Detailed Description

`#include <XEngine/VertexStruct.h>`

Namespaces

namespace	**XMath**

Classes

class	**XMath::Rectangle2**
	2D rectangle representation *More...*

RendererGL.h File Reference

Detailed Description

```
#include "IRenderer.h" #include "ITexture.h"
#include "Transform.h"
#include "DeformableGeometry.h"
#include <gl\gl.h>
#include <gl\glext.h>
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::RendererGL**
	OpenGL renderer. More...

RenderList.h File Reference

Detailed Description

```
#include <list> #include <boost/shared_ptr.hpp>
#include "Drawable.h"
#include "IRenderer.h"
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::RenderList**
	*Render list storing Drawables and calling proper **IRenderer** methods. More...*

Task.h File Reference

Detailed Description

```
#include <string> #include <map>
#include <deque>
```

Namespaces

namespace	**XFaceApp**

Classes

class	**XFaceApp::TaskDictionary** *Dictionary of available Task's. More...*
class	**XFaceApp::Task** ***Task** abstraction. More...*

TaskHandlerBase.h File Reference

Detailed Description

```
#include <string> #include <queue>
#include "Task.h"
```

Namespaces

namespace	**XFaceApp**

Classes

class	**XFaceApp::TaskHandlerBase** ***Task*** *handler base.* *More...*

Texture2D.h File Reference

Detailed Description

```
#include "ITexture.h"
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::Texture2D**
	2D texture mapping More...

TextureLoaderGL.h File Reference

Detailed Description

```
#include <list> #include "ITextureLoader.h"
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::TextureLoaderGL**
	Texture loader class for OpenGL. More...

TextureManager.h File Reference

Detailed Description

```
#include <map> #include <string>
#include "ITexture.h"
#include "ITextureLoader.h"
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::TextureManager**
	Textures are stored and managed here. More...

TgaFile.h File Reference

Detailed Description

```
#include "ITextureFile.h" #include <stdio.h>
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::TgaFile**
	TGA file loader for textures. More...

Transform.h File Reference

Detailed Description

```
#include <XMath/Vector3.h> #include <XMath/Matrix4.h>
#include <XMath/Quaternion.h>
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::Transform** ***Transform*** *node for scenegraph.* *More...*

Vector3.h File Reference

Detailed Description

```
#include <math.h>
```

Namespaces

namespace	**XMath**

Classes

class	**XMath::Vector3** *3D Vector class More...*

Defines

#define	**INFINITY** 3.402823466e+38F
	We define infinity value for avoiding divide by zero cases.

VertexStruct.h File Reference

Detailed Description

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::Vertex2D**
class	**XEngine::ColorRGBA**

Generated on Mon Aug 28 15:39:25 2006 for Xface Core Library by  *1.4.6-NO*

VRML1Loader.h File Reference

Detailed Description

```
#include "IModelLoader.h" #include "DeformableGeometry.h"
```

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::VRML1Loader**
	VRML1 Loader. More...

VRML97Loader.h File Reference

Detailed Description

#include "**IModelLoader.h**" #include "**IndexedFaceSet.h**"

Namespaces

namespace	**XEngine**

Classes

class	**XEngine::VRML97Loader**
	Loads VRML97 files. More...

XercesString.h File Reference

Detailed Description

```
#include <xercesc/util/XMLString.hpp>
```

Namespaces

namespace	**XFace**

Classes

class	**XFace::XercesString**
	Xerces string operations. *More...*

XLineSegment.h File Reference

Detailed Description

```
#include "Vector3.h"
```

Namespaces

namespace	**XMath**

Classes

class	**XMath::LineSegment3**
	Representation of a 3D line segment. More...

XMath.h File Reference

Detailed Description

```
#include "Ray3.h" #include "Vector3.h"
#include <XEngine/IndexedFaceSet.h>
```

Namespaces

namespace	**XMath**

XMLUtils.h File Reference

Detailed Description

```
#include <XFaceApp/Task.h> #include <XFaceApp/Notification.h>
#include <string>
```

Namespaces

namespace	**XFaceApp**

Classes

class	**XFaceApp::XMLUtils**
	XML related functions. More...

XTriangle.h File Reference

Detailed Description

```
#include "Vector3.h"
```

Namespaces

namespace	**XMath**

Classes

class	**XMath::Triangle3**
	3D triangle representation More...

Xface Core Library Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

XEngine::MorphChannel::destroy_functor	
XEngine::MorphChannel::smooth_functor	
XEngine::BmpFile	*BMP file loader for textures*
XEngine::ColorRGBA	
XEngine::DeformableGeometry	*Deformation enabled geometry class*
XEngine::Drawable	*Stores the information for drawable objects*
XEngine::Entity	*Represents an entity visible in the scene, includes drawables, ai, etc.*
XEngine::Geometry	*Defines interface for the mesh geometry*
XEngine::IModelLoader	*Base class for the scene graph loaders*
XEngine::IndexedFaceSet	*Indexed mesh geometry implemented*
XEngine::IRenderer	*Interface for renderer*
XEngine::ITexture	*Base class for texture mapping*
XEngine::ITextureFile	*Interface for image file loaders for textures*
XEngine::ITextureLoader	*Base class for texture loading*
XEngine::ITimedController	*ABS for timed controllers*
XEngine::ITimer	*Timer interface*
XEngine::KeyframeInterpolator	*Keyframe interpolator*
XEngine::MeshInfo	*3D Meshe info*
XEngine::MeshManager	*Meshes are stored and managed here*
XEngine::ModelFileFactory	*3D Meshes are loaded from here*
XEngine::MorphChannel::dump_functor	
XEngine::MorphController	*Morph targets controller*
XEngine::NamedObj	*Named classes base*
XEngine::OBJLoader	*Loads Wavefront OBJ files for the scene graph*
XEngine::RendererGL	*OpenGL renderer*
XEngine::RenderList	*Render list storing Drawables and calling proper **IRenderer** methods*
XEngine::RenderManager	*Manages creation of renderer, storage of drawables to be rendered and rendering them*
XEngine::Texture2D	*2D texture mapping*

XEngine::TextureLoaderGL	*Texture loader class for OpenGL*
XEngine::TextureManager	*Textures are stored and managed here*
XEngine::TgaFile	*TGA file loader for textures*
XEngine::Transform	***Transform*** *node for scenegraph*
XEngine::Vertex2D	
XEngine::VRML1Loader	*VRML1 Loader*
XEngine::VRML97Loader	*Loads VRML97 files*
XFace::FaceBase	*Main* ***XFace*** *class*
XFace::FAPFile	*MPEG-4 FAPs loader from disk*
XFace::FAPU	*MPEG4* ***FAPU*** *info is in here*
XFace::FDP	*MPEG4* ***FDP*** *data as a whole*
XFace::FDPFile	*MPEG-4* ***FDP*** *loader/writer from/to disk*
XFace::FDPItem	*MPEG4* ***FDP*** *data*
XFace::FDPLoader	***FDP*** *Loader*
XFace::IFapStream	*Interface for MPEG-4 FAPs streaming*
XFace::IInfluenceCalculator	*Interface for defining the weights in the zone of influence for deformation*
XFace::InfluenceCalculatorMaker	*InfluenceCalculator pluggable factory maker*
XFace::RaisedCosInfluence	*Raised cosine deformation base*
XFace::RaisedCosInfluenceSph	*Raised cosine deformation as a sphere*
XFace::RaisedCosInfluenceSphMaker	*Raised cosine influence maker*
XFace::RaisedCosInfluenceWaveX	*Raised cosine deformation as a wave in X direction*
XFace::RaisedCosInfluenceWaveXMaker	*Raised cosine influence maker*
XFace::RaisedCosInfluenceWaveY	*Raised cosine deformation as a wave in Y direction*
XFace::RaisedCosInfluenceWaveYMaker	*Raised cosine Influence maker*
XFace::RaisedCosInfluenceWaveZ	*Raised cosine deformation as a wave in Z direction*
XFace::RaisedCosInfluenceWaveZMaker	*Raised cosine Influence maker*
XFace::XercesString	*Xerces string operations*
XFaceApp::ApplicationBase	*Base for application dev classes*
XFaceApp::ISound	*Sound player interface*
XFaceApp::ModelCamera	*Camera class*
XFaceApp::Notification	***Notification*** *message to server for remote communication*
XFaceApp::Task	***Task*** *abstraction*

XFaceApp::TaskDictionary	*Dictionary of available Task's*
XFaceApp::TaskHandlerBase	***Task*** *handler base*
XFaceApp::XMLUtils	*XML related functions*
XMath::LineSegment3	*Representation of a 3D line segment*
XMath::Matrix4	*4x4 Matrix class for float*
XMath::Quaternion	***Quaternion*** *class*
XMath::Ray3	*3D ray representation*
XMath::Rectangle2	*2D rectangle representation*
XMath::Triangle3	*3D triangle representation*
XMath::Vector3	*3D Vector class*

XEngine::MorphChannel::destroy_functor

XEngine::MorphChannel::destroy_functor Struct Reference

`#include <MorphChannel.h>`

List of all members.

Detailed Description

destroys MorphTarget objects (pointers)

The documentation for this struct was generated from the following file:

- MorphChannel.h

XEngine::MorphChannel::smooth_functor

XEngine::MorphChannel::smooth_functor Struct Reference

`#include <MorphChannel.h>`

List of all members.

Detailed Description

smooths MorphTarget objects with the "Rest" keyframe

The documentation for this struct was generated from the following file:

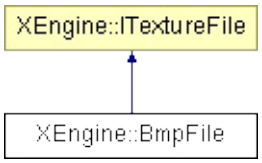
- MorphChannel.h

XEngine::BmpFile

XEngine::BmpFile Class Reference
[XEngine]

```
#include <BmpFile.h>
```

Inheritance diagram for XEngine::BmpFile:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
April 2003

Loads BMP files as texture map.

Public Member Functions

bool	**load** (const std::string &filename)

Member Function Documentation

bool XEngine::BmpFile::load (const std::string & *filename*) [virt

Loads the BMP file to memory (acces data using **ITextureFile** base class members).

Note:

Uses glaux library for the moment. It doesn't come with devcpp/gcc so do not forget to add it as a dependency.

Implements **XEngine::ITextureFile**.

The documentation for this class was generated from the following files:

- **BmpFile.h**
- BmpFile.cpp

XEngine::ColorRGBA

XEngine::ColorRGBA Class Reference
[XEngine]

```
#include <VertexStruct.h>
```

List of all members.

Detailed Description

RGBA color structure

The documentation for this class was generated from the following file:

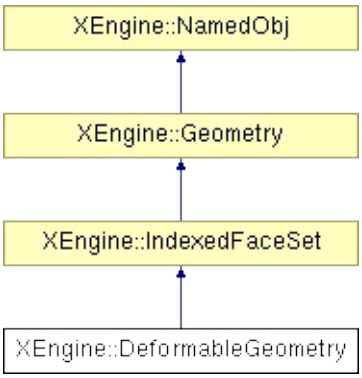
- **VertexStruct.h**

XEngine::DeformableGeometry

XEngine::DeformableGeometry Class Reference [XEngine]

`#include <DeformableGeometry.h>`

Inheritance diagram for XEngine::DeformableGeometry:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
Sept 2004

Deformation is done on this **Geometry** derived class.

Public Member Functions

void	**setVertices** (const std::vector< **Vector3** > &vertices)
void	**update** (const std::vector< float > &aps)

Member Function Documentation

void XEngine::DeformableGeometry::setVertices (const std::vecto

Note:

when sth rendered in keyframe mode, if after we try to load some fap, lips do not move anymore. reason: clearInfluences call in DeformableGeometry::setVertices cleared the weights. I just cannot remember why this line is there, when removed things seem to work fine now. May have side effects though.

Reimplemented from **XEngine::Geometry**.

void XEngine::DeformableGeometry::update (const std::vector< fl

Todo:

computeVertexNormals() ??

The documentation for this class was generated from the following files:

- **DeformableGeometry.h**
- DeformableGeometry.cpp

XEngine::Drawable

XEngine::Drawable Class Reference
[XEngine]

```
#include <Drawable.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
May 2003

Stores the information about the drawables that can be rendered. We store the name of the Mesh and Textures (for the moment, others in the future) as std::string member variables. This class acts only as an alias that stores the references (names) of the real data.

We store the **Drawable** instances for head, eyes, etc in **RenderList** object. Renderer (or derived **RendererGL**) class use the information in **Drawable** class and retrieves the appropriate data and renders them.

Public Member Functions

Drawable *	**clone** (bool duplicateData=true) const
void	**enableTexture** (bool mode) *Enable texture mapping on/off for this drawable.*
bool	**isTextureOn** () const *Query if texture mapping is enabled.*
	Drawable () *Default CTor.*
	~Drawable () *DTor (not virtual!).*
void	**setMeshName** (const std::string &MeshName) *Sets the name of associated mesh.*
const std::string &	**getMeshName** () const *Retrieves the name of associated mesh.*
void	**setTexName** (const std::string id, unsigned short layer=0)
const std::string &	**getTexName** (unsigned short layer=0) const

Member Function Documentation

Drawable* XEngine::Drawable::clone (bool *duplicateData* = `true`)

clones the drawable object instance. duplicates data by creating a new **Geometry** (**DeformableGeometry**) object and copying the data content by calling **Geometry::copyFrom** method if duplicateData parameter is true.

const std::string& XEngine::Drawable::getTexName (unsigned sho

Retrieves the name of the texture for each layer. Default parameter use the first texture unit.

Parameters:

layer Order of texture unit to be used for this texture.

void XEngine::Drawable::setTexName (const std::string *id*, unsigned short *layer* = `0`) `[inline]`

Sets the name of the texture for each layer. Default parameter use the first texture unit.

Parameters:

id Name of the texture.

layer Order of texture unit to be used for this texture. Defaulted to 0 and cannot be greater than 3.

The documentation for this class was generated from the following file:

- **Drawable.h**

XEngine::Entity

XEngine::Entity Class Reference

`#include <Entity.h>`

List of all members.

The documentation for this class was generated from the following files:

- **Entity.h**
- Entity.cpp

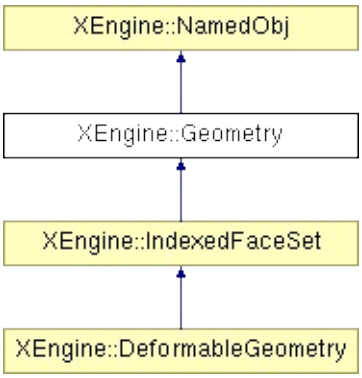
Generated on Mon Aug 28 15:39:25 2006 for Xface Core Library by  *1.4.6-NO*

XEngine::Geometry

XEngine::Geometry Class Reference
[XEngine]

`#include <Geometry.h>`

Inheritance diagram for XEngine::Geometry:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
May 2003

Base class for the triangle mesh. Stores the vertices (not indices) and other related data. Every IMesh object is named and stored in **MeshManager** to be used through its name. The same name is assigned to **Drawable** instance and the make a couple for the mesh, **MeshManager** stores the IMesh instances, and **RenderList** stores the Drawables and Renderer renders IMesh instances by retrieving them using the registered names in **Drawable** instance.

IMesh objects are created and stored inside **IModelLoader** (or derived) instances, and stored in **MeshManager** class.

Here is a typical usage in **IModelLoader**;

```
        IndexedFaceSet* m_pMesh = new IndexedFaceSet("TestMesh"); // Cr
        // Set a unique name in ctor
        m_pMesh->setVertices(m_Vertices);       // Set the vertices and
        m_pMesh->setIndices(m_Indices);         // Set the indices
        m_pMesh->createMesh();                  // You SHOULD call this

        // MeshManager is a singleton class.
        MeshManager::getInstance()->registerMesh(m_pMesh); // Add the m
```

Then when you need to use that mesh in a different place (e.g. in Renderer);

```
        const IndexedGFaceSet* test = MeshManager::getInstance()->getMe
```

Note:
Texture coordinates, normals and vertices are stored in seperate

arrays. Maybe we can consider using a single array (interleaved) later on. And also currently only one texture layer is supported. For multitexturing we might add more arrays for texture coordinates in the future versions.

Public Member Functions

virtual **Geometry** &	**copyFrom** (const **Geometry** &rhs)
virtual void	**setVertices** (const **Vector3** *pVert, unsigned int size)
virtual void	**setVertices** (const std::vector< **Vector3** > &vertices)
virtual void	**setNormals** (const std::vector< **Vector3** > &normals)
virtual void	**setNormals** (const **Vector3** *pNorm, unsigned int size)
virtual void	**setTexCoords** (const **Vertex2D** *pTex, unsigned int size)
virtual void	**setTexCoords** (const std::vector< **Vertex2D** > &texCoords)
const std::vector< **Vector3** > &	**getVertices** () const *Accessor for vertices (x, y, z).*
const std::vector< **Vector3** > &	**getNormals** () const *Accessor for normals.*
const std::vector< **Vertex2D** > &	**getTexCoords** () const *Accessor for texture coordinates.*
size_t	**getVertexCount** () const *Accessor for vertex count.*

Static Public Member Functions

static **Vector3**	**computeFaceNormal** (const **Vector3** &p1, const **Vector3** &p2, const **Vector3** &p3)

Member Function Documentation

Vector3 XEngine::Geometry::computeFaceNormal (const Vector3, const Vector3, const Vector3) [static]

Computes the facenormal by crossing two vectors defined by (p2-p1) and (p3-p1). The parameters are assumed to be passed in counter-clockwise order.

Geometry & XEngine::Geometry::copyFrom (const Geometry & *rh*

We do NOT override the = operator but implement a new method, because in some cases, we only want to copy the data, not the name of the **Geometry**.

void XEngine::Geometry::setNormals (const Vector3 * *pNorm*, unsigned int *size*) [virtual]

Todo:
check back here

void XEngine::Geometry::setNormals (const std::vector< Vector3

Vertex normals are set through here.

void XEngine::Geometry::setTexCoords (const std::vector< Vertex

Texture coordinates are set through here.

void XEngine::Geometry::setTexCoords (const Vertex2D * *pTex*, unsigned int *size*) [virtual]

Texture coordinates are set through here.

void XEngine::Geometry::setVertices (const std::vector< Vector3 >

Vertices are set through here.

Reimplemented in **XEngine::DeformableGeometry**.

void XEngine::Geometry::setVertices (const Vector3 * *pVert*, unsigned int *size*) [virtual]

Todo:
check back here

The documentation for this class was generated from the following files:

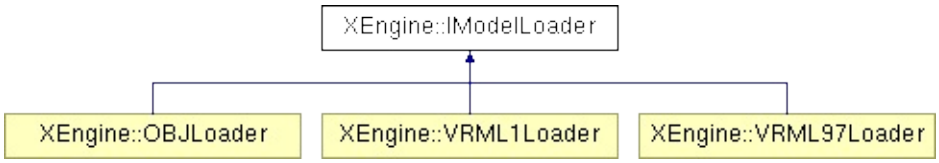
- **Geometry.h**
- Geometry.cpp

XEngine::IModelLoader

XEngine::IModelLoader Class Reference [XEngine]

```
#include <IModelLoader.h>
```

Inheritance diagram for XEngine::IModelLoader:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
March 2003

This class should be the base for the scene graph loaders. Currently, has only one pure virtual function to define the interface to load scenes.

With this abstract base class, we ensure future extendibility for various 3D model/scene formats.

Public Member Functions

virtual std::list< boost::shared_ptr< **Drawable** > >	**loadModel** (const std::string &filename, const std::string &dir="./")=0

Member Function Documentation

virtual std::list<boost::shared_ptr<Drawable> > XEngine::IModelLo

Pure virtual function for loader.

Parameters:

filename	Name of the model file.
dir	Path of the model file.

Returns:

the list of the names of meshes loaded.

Implemented in **XEngine::OBJLoader**, **XEngine::VRML1Loader**, and **XEngine::VRML97Loader**.

The documentation for this class was generated from the following file:

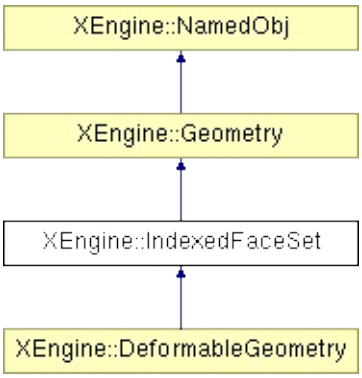
- **IModelLoader.h**

XEngine::IndexedFaceSet

XEngine::IndexedFaceSet Class Reference [XEngine]

`#include <IndexedFaceSet.h>`

Inheritance diagram for XEngine::IndexedFaceSet:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
May 2003

Implements the indexed face set from the **Geometry** class. Each polygon is a triangle in the geometry, if not they are corrected automatically when the indices are set by the loader.

See base class documentation for use and more info.

See also:
Geometry

Public Member Functions

void	**computeVertexNormals** ()
IndexedFaceSet &	**copyFrom** (const **IndexedFaceSet** &rhs) *overriding base class virtual method*
size_t	**getIndexCount** () const *Returns the number of indices.*
const unsigned short *	**getIndices** () const *Accessor for the indices to the triangles.*
void	**setIndices** (const std::vector< std::vector< unsigned short > > &indices)

Member Function Documentation

void XEngine::IndexedFaceSet::computeVertexNormals ( )

Computes the vertex normals if the mesh has an indexed structure. For computing the vertex normals, simply the face normals for each polygon is found by crossing two edges of the polygon, then corresponding vertices' normals are summed up with the face normal. At last, all the vertex normals are normalized. Might be implemented faster, but since this step is done only once in the initialization of the mesh, it is not a bottleneck. However, for dynamically changing geometry, some optimizations can be considered.

void XEngine::IndexedFaceSet::setIndices (const std::vector< std

Indices are set here. If a face is not a triangle it is divided into triangles so that the final geometry contains only triangles (which is a Good Thing!).

Parameters:

indices	A vector of vector of unsigned short values representing indices to vertices. Inner vector represents each face (polygon) and the outer vector is the whole colection of faces.

The documentation for this class was generated from the following files:

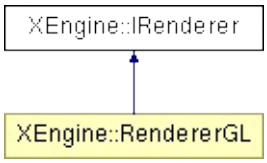
- **IndexedFaceSet.h**
- IndexedFaceSet.cpp

XEngine::IRenderer

XEngine::IRenderer Class Reference
[XEngine]

```
#include <IRenderer.h>
```

Inheritance diagram for XEngine::IRenderer:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
May 2003

Abstract base class for all type of renderers. This class ensures the future development paths for different API's like OpenGL, DirectX, software, etc..

Public Member Functions

virtual void	**render** (**Drawable** *const pDrawable) const =0
	*Pure virtual method for rendering a **Drawable** object.*

The documentation for this class was generated from the following file:

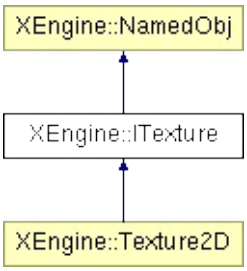
- **IRenderer.h**

XEngine::ITexture

XEngine::ITexture Class Reference
[XEngine]

`#include <ITexture.h>`

Inheritance diagram for XEngine::ITexture:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0 (August 2002)
1.1 (Jan 2003)

Serves as a base class for texture mapping. Stores the texture name and ID, **ITexture** instances are created in **ITextureLoader** interface and registered to **TextureManager** using **TextureManager::load()** member.

Each instance should be named whit a unique string and registered to **TextureManager** for consistent behaviour.

See also:
Texture2D, **ITextureLoader**, **TextureLoaderGL**, **TextureManager**

Public Types

enum	**TEXTURETYPE** { **TEXTURE2D** }

Public Member Functions

virtual **TEXTURETYPE**	**getTextureType** () const =0 *Returns the type of texture.*
const unsigned int	**getTextureID** () const *Returns the texture id attached to this object.*
	ITexture (const std::string &name) *Constructor.*
virtual	**~ITexture** () *Destructor.*

Member Enumeration Documentation

enum XEngine::ITexture::TEXTURETYPE

Enumeration for the types of textures, currently there is only one, but later, we will need 1D and 3D as well perhaps.

Enumerator:

TEXTURE2D 2D Texture type identifier

The documentation for this class was generated from the following file:

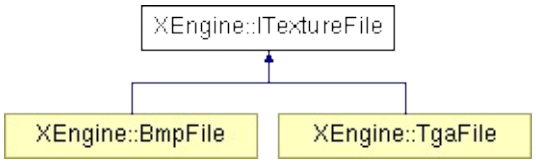
- **ITexture.h**

XEngine::ITextureFile

XEngine::ITextureFile Class Reference
[XEngine]

```
#include <ITextureFile.h>
```

Inheritance diagram for XEngine::ITextureFile:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
April 2003

Interface for texture loading. Derived classes implement the **load()** member function in order to support loading of various image file formats.

Public Member Functions

unsigned char *	**getData** () *Accessor for loaded data.*
int	**getNBits** () const *Accessor for number of bits per pixel.*
int	**getWidth** () const *Accessor for width of the image.*
int	**getHeight** () const *Accessor for height of the image.*
virtual bool	**load** (const std::string &filename)=0 *Pure virtual interface method for loading files. Main entry point.*

Protected Attributes

int	**m_Width** *Width of the image.*
int	**m_Height** *Height of the image.*
int	**m_nBits** *Number of bits per pixel.*
unsigned char *	**m_pData** *Image data here.*

The documentation for this class was generated from the following file:

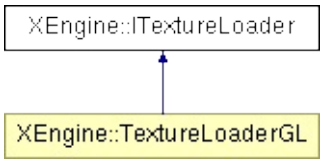
- **ITextureFile.h**

XEngine::ITextureLoader

XEngine::ITextureLoader Class Reference [XEngine]

`#include <ITextureLoader.h>`

Inheritance diagram for XEngine::ITextureLoader:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
August 2002

Serves as a base class for texture loading. Note that, loading a texture is an API (D3D/OpenGL) dependant operation. To ensure API independence, we use this pure abstract base class.

Public Member Functions

virtual void	**unLoad** (const **ITexture** *pTexture)=0 *UnLoads a texture attached to an **ITexture** object.*
virtual bool	**load** (const std::string &filename, **ITexture** *pTexture)=0 *Loads the texture.*
	ITextureLoader () *Default constructor.*
virtual	**~ITextureLoader** () *Destructor.*

The documentation for this class was generated from the following file:

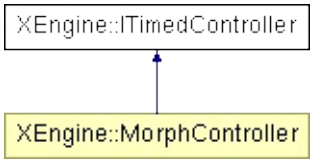
- **ITextureLoader.h**

XEngine::ITimedController

XEngine::ITimedController Class Reference

`#include <ITimedController.h>`

Inheritance diagram for XEngine::ITimedController:



List of all members.

The documentation for this class was generated from the following file:

- **ITimedController.h**

XEngine::ITimer

XEngine::ITimer Class Reference
[XEngine]

```
#include <ITimer.h>
```

List of all members.

Detailed Description

Author:

Koray Balci Provides timer interface for animation synchronization.

The documentation for this class was generated from the following file:

- **ITimer.h**

XEngine::KeyframeInterpolator

XEngine::KeyframeInterpolator Class Reference

`#include <KeyframeInterpolator.h>`

List of all members.

The documentation for this class was generated from the following files:

- **KeyframeInterpolator.h**
- KeyframeInterpolator.cpp

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XEngine::MeshInfo

XEngine::MeshInfo Struct Reference
[XEngine]

```
#include <ModelFileFactory.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
Nov 2003

Stores the information about the mesh file, both the list of drawables and file info for file name, path and format.

The documentation for this struct was generated from the following file:

- **ModelFileFactory.h**

XEngine::MeshManager

XEngine::MeshManager Class Reference [XEngine]

```
#include <MeshManager.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
May 2003

Meshes (**IndexedFaceSet** instances) are stored and managed through here. All the meshes are stored in a hash map (currently) with a string representing the name as the key of the hash. **MeshManager** is a singleton class.

Here is a typical usage in **IModelLoader**;

```
        IndexedFaceSet* m_pMesh = new IndexedFaceSet; // Create an inst
        ... // Initialization, etc..

        // MeshManager is a singleton class.
        MeshManager::getInstance()->registerMesh(m_pMesh); // Add the m
```

And here is how we access the loaded Mesh (from **RendererGL**);

```
        Drawable* pDrawable; // Assume it is initialized correctly.
        ...
        std::string MeshName = pDrawable->getMeshName();
        const DeformableGeometry* pMesh = MeshManager::getInstance()->g
```

Public Member Functions

void	**destroyAll** ()
void	**registerMesh** (**DeformableGeometry** *pMesh)
DeformableGeometry *const	**getMesh** (const std::string &name) const

Static Public Member Functions

static void	**releaseInstance** ()
static **MeshManager** *	**getInstance** ()

Member Function Documentation

void XEngine::MeshManager::destroyAll ( )

Destroys all of the meshes in the storage.

MeshManager * XEngine::MeshManager::getInstance ( ) [static]

Singleton class accessor for the one and only instance of this class.

DeformableGeometry *const XEngine::MeshManager::getMesh (co

Retrieves a mesh with the name passed.

Returns:
0 if unsucessful, a valid pointer to IMesh if successful.

void XEngine::MeshManager::registerMesh (DeformableGeometry

Registers a mesh object to the storage.

void XEngine::MeshManager::releaseInstance ( ) [static]

Releases the one and only instance.

The documentation for this class was generated from the following files:

- MeshManager.h
- MeshManager.cpp

XEngine::ModelFileFactory

XEngine::ModelFileFactory Class Reference [XEngine]

```
#include <ModelFileFactory.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
Nov 2003

This class is used to keep track of which mesh files are currently loaded through **MeshManager** and in memory. It is a Monolithic design pattern impl where all members are static.

Todo:
I guess this is only necessary for XFaceEd, so we might consider moving this class there.

Static Public Member Functions

static const **MeshInfo** *	**getMeshInfo** (const std::string &meshName) *returns the list of **MeshInfo** for the files already loaded.*
static std::list< boost::shared_ptr< **Drawable** > >	**loadModelFile** (const std::string &filename, const std::string &path="./")
static **MeshInfo**	**unloadModelFile** (const std::string &filename)
static const **MeshInfo** *	**isFileLoaded** (const std::string &filename)
static void	**unloadAllFiles** ()

Member Function Documentation

const MeshInfo * XEngine::ModelFileFactory::isFileLoaded (const

Checks is the file is already loaded or not.

Returns:
0 pointer if it is not loaded, pointer to **MeshInfo** already loaded otherwise.

std::list< boost::shared_ptr< Drawable > > XEngine::ModelFileFact

Load the mesh from here.

void XEngine::ModelFileFactory::unloadAllFiles ( ) [static]

UnLoads all the mesh files from memory.

MeshInfo XEngine::ModelFileFactory::unloadModelFile (const std

UnLoad a single mesh file from here. filenames should be unique regardless of the path they have. i.e. you cannot have a mesh file with the same file name located at different paths!

The documentation for this class was generated from the following files:

- **ModelFileFactory.h**
- ModelFileFactory.cpp

XEngine::MorphChannel::dump_functor

XEngine::MorphChannel::dump_functor Struct Reference

`#include <MorphChannel.h>`

List of all members.

Detailed Description

dumps MorphTarget objects

The documentation for this struct was generated from the following file:

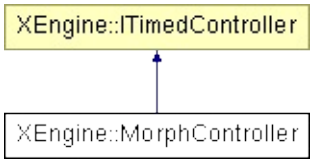
- MorphChannel.h

XEngine::MorphController

XEngine::MorphController Class Reference

`#include <MorphController.h>`

Inheritance diagram for XEngine::MorphController:



List of all members.

Static Public Member Functions

static **MorphController** *	**getInstance** ()
	singleton access

The documentation for this class was generated from the following files:

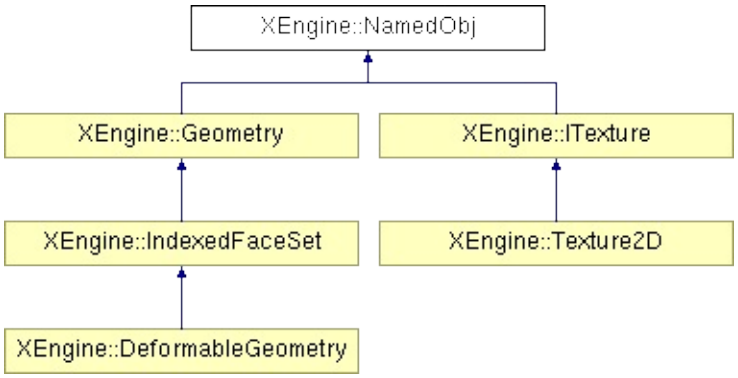
- **MorphController.h**
- MorphController.cpp

XEngine::NamedObj

XEngine::NamedObj Class Reference
[XEngine]

`#include <NamedObj.h>`

Inheritance diagram for XEngine::NamedObj:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
Nov 2003

A class that serves so little.. Thought it would be a good idea to not repeat adding get/set Name stuff everytime it is necessary for a class, however, not so sure anymore, might be breaking some holly OO rule!

Public Member Functions

void	**setName** (const std::string &name) *Accessor for Name prop.*
const std::string &	**getName** () const *Accessor for Name prop.*

Protected Member Functions

	NamedObj (const std::string &name) *Thou shall not create an instance of this class.*
virtual	**~NamedObj** (void) *I know when protected, no need to make it virtual, so what?*

The documentation for this class was generated from the following file:

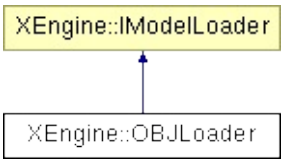
- **NamedObj.h**

XEngine::OBJLoader

XEngine::OBJLoader Class Reference
[XEngine]

```
#include <OBJLoader.h>
```

Inheritance diagram for XEngine::OBJLoader:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
April 2003

Loads Wavefront OBJ files, creates the scene graph nodes, and attaches them. Uses **MeshManager** class to register the meshes loaded.

Public Member Functions

std::list< boost::shared_ptr< **Drawable** > >	**loadModel** (const std::string &filename, const std::string &dir="./")

Member Function Documentation

std::list< boost::shared_ptr< Drawable > > XEngine::OBJLoader::lo

Pure virtual function for loader.

Parameters:

filename	Name of the model file.
dir	Path of the model file.

Returns:

the list of the names of meshes loaded.

Implements **XEngine::IModelLoader**.

The documentation for this class was generated from the following files:

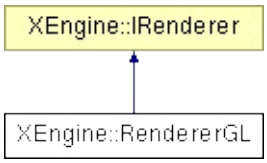
- **OBJLoader.h**
- OBJLoader.cpp

XEngine::RendererGL

XEngine::RendererGL Class Reference
[XEngine]

```
#include <RendererGL.h>
```

Inheritance diagram for XEngine::RendererGL:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
May 2003

OpenGL implementation of the **IRenderer** interface. Using OpenGL API, rendering is handled.

Public Member Functions

void	**render** (**Drawable** *const pDrawable) const

Member Function Documentation

void XEngine::RendererGL::render (Drawable *const *pDrawable*

Renders a **Drawable** instance using OpenGL.

Implements **XEngine::IRenderer**.

The documentation for this class was generated from the following files:

- **RendererGL.h**
- RendererGL.cpp

XEngine::RenderList

XEngine::RenderList Class Reference
[XEngine]

```
#include <RenderList.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.1

Date:
May 2003 modified : added boost::shared_ptr support + functors

This class keeps a list of pointers to **Drawable** instances. Once **renderList()** method is called, contents of the list are sent to **IRenderer** sequentially.

Public Member Functions

void	**addDrawable** (boost::shared_ptr< **Drawable** > item) *Adds a **Drawable** pointer to the list.*
void	**addDrawables** (const std::list< boost::shared_ptr< **Drawable** > > &dr) *Adds a **Drawable** list to the list.*
void	**removeDrawable** (const std::string &name) *Removes a **Drawable** with given name.*
void	**clearList** () *Clears the list.*
void	**renderList** (boost::shared_ptr< **IRenderer** > pRenderer) const *Sends the contents of the list to **IRenderer** sequentially.*

The documentation for this class was generated from the following files:

- **RenderList.h**
- RenderList.cpp

XEngine::RenderManager

XEngine::RenderManager Class Reference

```
#include <RenderManager.h>
```

List of all members.

Public Member Functions

void	**update** ()

Member Function Documentation

void XEngine::RenderManager::update ( )

does nothing now, can be used for sorting the render list in the future.

The documentation for this class was generated from the following files:

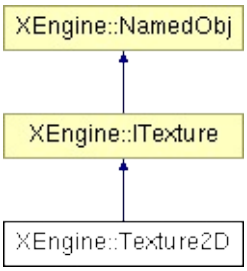
- RenderManager.h
- RenderManager.cpp

XEngine::Texture2D

XEngine::Texture2D Class Reference
[XEngine]

```
#include <Texture2D.h>
```

Inheritance diagram for XEngine::Texture2D:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
August 2002

2D (ordinary) texture mapping parameters are stored here.

Here is how it is used in **RendererGL** using OpenGL;

```
// Get a pointer to base class for the texture;
Drawable* pDrawable; // A valid instance
...
const ITexture* pTexture = TextureManager::getInstance(
// if it is a 2D texture
if(pTexture->getTextureType() == ITexture::TEXTURE2D)
{
        glEnable(GL_TEXTURE_2D);
        glBindTexture(GL_TEXTURE_2D, pTexture->getTextu
}
```

Public Member Functions

TEXTURETYPE	**getTextureType** () const *Returns the type of the texture. (TEXTURE2D).*
	Texture2D (const std::string &name) *Default Constructor.*
virtual	**~Texture2D** () *Destructor.*

The documentation for this class was generated from the following file:

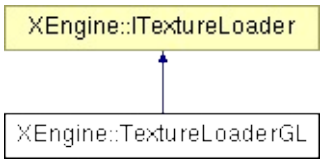
- **Texture2D.h**

XEngine::TextureLoaderGL

XEngine::TextureLoaderGL Class Reference [XEngine]

```
#include <TextureLoaderGL.h>
```

Inheritance diagram for XEngine::TextureLoaderGL:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
August 2002

Loads the textures from files (later might create textures dynamically) using OpenGL function calls and stores the texture list. Deletes the textures on exit. An STL list is used to store the textures.

Note:
Do NOT destroy the instance of this class if you still want to use the textures created by it. It DELETEs all the textures created by this instance.

In the framework, it is used in **TextureManager** class as a member variable. So, if possible use **TextureManager** to load/unload and manage your textures.

Public Member Functions

void	**unLoad** (const **ITexture** *pTexture) *Unloads the OpenGL texture from the **ITexture** instance.*
bool	**load** (const std::string &filename, **ITexture** *pTexture) *Loads the texture with the filename to the **ITexture** instance.*
	TextureLoaderGL () *Default constructor.*
virtual	**~TextureLoaderGL** ()

Constructor & Destructor Documentation

XEngine::TextureLoaderGL::~TextureLoaderGL ( ) [virtual]

IMPORTANT: Clears the texture list and deletes the textures if there are any.

Member Function Documentation

**bool XEngine::TextureLoaderGL::load (const std::string & *filenam*
ITexture * *pTextu*
) [virtual]**

Loads a texture from a file and attaches it to an **ITexture** object.

Parameters:

filename	Path to an image file to load the texture.
pTexture	A valid pointer to an existing **ITexture** instance to attach the loaded texture.

Note:
Currently only bitmap (using gluaux library) and tga files are supported.

Implements **XEngine::ITextureLoader**.

void XEngine::TextureLoaderGL::unLoad (const ITexture * *pTextu*

Unloads a texture attached to a Texture object.

Implements **XEngine::ITextureLoader**.

The documentation for this class was generated from the following files:

- **TextureLoaderGL.h**
- TextureLoaderGL.cpp

XEngine::TextureManager

XEngine::TextureManager Class Reference [XEngine]

```
#include <TextureManager.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
May 2003

Textures (**ITexture** instances) are loaded, stored and managed through here. All the textures are stored in a hash map (currently) with a string representing the name as the key of the hash. **TextureManager** is a singleton class.

Here is a sample usage for loading a texture;

```
TextureManager::getInstance()->load("..\\john_lo_obj02.
// this is optional but good to do, associate the textu
Drawable* pDrawable = new Drawable;
pDrawable->setTexName("john_lo_obj00");
```

And here is how we access the loaded texture (from **RendererGL**);

```
const ITexture* pTexture = TextureManager::getInstance(
```

Public Member Functions

const **ITexture** *	**load** (const std::string &filename, const std::string &texname)
void	**unLoad** (const **ITexture** *pTexture)
void	**unLoad** (const std::string &name)
void	**destroyAll** ()
void	**registerTexture** (const **ITexture** *pTexture)
void	**unregisterTexture** (const **ITexture** *pTexture)
const **ITexture** *	**getTexture** (const std::string &name) const

Static Public Member Functions

static void	**releaseInstance** ()
static **TextureManager** *	**getInstance** ()

Member Function Documentation

void XEngine::TextureManager::destroyAll ( )

Destroys all of the textures in the storage.

TextureManager * XEngine::TextureManager::getInstance ( ) [stat

Singleton class accessor for the one and only instance of this class.

const ITexture * XEngine::TextureManager::getTexture (const std::

Retrieves a texture with the name passed.

Returns:
0 if unsucessful, a valid pointer to **ITexture** if successful.

const ITexture * XEngine::TextureManager::load (const std::string
const std::string
)

Loads a texture from a file, creates an **ITexture** instance, binds the texture to it, and registers it to texture storage.

Parameters:

filename	Name of the image file to be used as a texture.
texname	Name of the texture object to be created and binded to texture.

void XEngine::TextureManager::registerTexture (const ITexture *

Registers a texture object to the storage.

void XEngine::TextureManager::releaseInstance ( ) [static]

Releases the one and only instance.

void XEngine::TextureManager::unLoad (const std::string & *name*

Unloads the texture completely and removes it from the storage.

Parameters:

name Name of the texture object in the storage.

void XEngine::TextureManager::unLoad (const ITexture * *pTextur*

Unloads the texture completely and removes it from the storage.

Parameters:

pTexture A valid, existant **ITexture** instance.

void XEngine::TextureManager::unregisterTexture (const ITexture

Removes a texture from the storage.

The documentation for this class was generated from the following files:

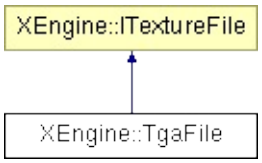
- **TextureManager.h**
- TextureManager.cpp

XEngine::TgaFile

XEngine::TgaFile Class Reference
[XEngine]

```
#include <TgaFile.h>
```

Inheritance diagram for XEngine::TgaFile:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
April 2003

Loads TGA files for textures.

Public Member Functions

bool	**load** (const std::string &filename)

Member Function Documentation

bool XEngine::TgaFile::load (const std::string & *filename*) [virtu

Loads the TGA file to memory (acces data using **ITextureFile** base class members).

Implements **XEngine::ITextureFile**.

The documentation for this class was generated from the following files:

- **TgaFile.h**
- TgaFile.cpp

XEngine::Transform

XEngine::Transform Class Reference
[XEngine]

```
#include <Transform.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Date:
April 2004

Represents the transformation node of the scenegraph by storing translation and rotation information.

See also:
Drawable, **Geometry**

Public Member Functions

Quaternion	**getRotation** () const *Accessor.*
Vector3	**getTranslation** () const *Accessor.*
void	**setTranslation** (const **Vector3** &trans) *Accessor.*
void	**setTranslation** (float x, float y, float z) *Accessor.*
void	**setRotation** (const **Quaternion** &qRot) *Accessor.*
void	**setScale** (const **Vector3** &scale) *Accessor.*

The documentation for this class was generated from the following files:

- **Transform.h**
- Transform.cpp

XEngine::Vertex2D

XEngine::Vertex2D Class Reference
[XEngine]

```
#include <VertexStruct.h>
```

List of all members.

Detailed Description

2D Vertex used for texture coordinates

The documentation for this class was generated from the following file:

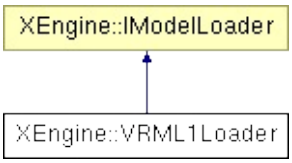
- **VertexStruct.h**

XEngine::VRML1Loader

XEngine::VRML1Loader Class Reference [XEngine]

`#include <VRML1Loader.h>`

Inheritance diagram for XEngine::VRML1Loader:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
Feb 2004

Loads 3D VRML1 files, creates the meshes and registers them through **MeshManager**.

Public Member Functions

std::list< boost::shared_ptr< **Drawable** > >	**loadModel** (const std::string &filename, const std::string &dir="./")
	Overriden base class method for loading scene.

The documentation for this class was generated from the following files:

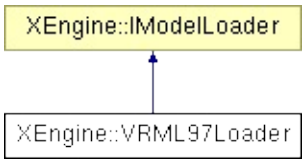
- **VRML1Loader.h**
- VRML1Loader.cpp

XEngine::VRML97Loader

XEngine::VRML97Loader Class Reference [XEngine]

```
#include <VRML97Loader.h>
```

Inheritance diagram for XEngine::VRML97Loader:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
April 2004

Loads VRML97 files, creates the meshes and registers them through **MeshManager**. Code that does the job, saves the day, but not safe, lots of pitfalls ignored, huge possibility of crash and room for improvement.

Todo:
Not finished yet, need to implement full scene graph struct for proper use.

Public Member Functions

std::list< boost::shared_ptr< **Drawable** > >	**loadModel** (const std::string &filename, const std::string &dir="./")
	Overriden base class method for loading scene.

The documentation for this class was generated from the following files:

- **VRML97Loader.h**
- VRML97Loader.cpp

XFace::FaceBase

XFace::FaceBase Class Reference
[XFace]

```
#include <FaceBase.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
June 2003

Main entry point for **XFace** library related operations. Only exposing this class to users should be sufficient. Pass the **FDP** configuration file for initialization, then call update and render when needed.

```
FaceBase myFace;
myFace.init(FDPFilename, FDPPath); // load an FDP file

// in a loop of all FAP data
myFace.update(FAPData);
```

Note:
Alternatively, you can use **XFaceApp** module and use XML tasks as messages (see related documentation) and forget about this class and all other **XFace** classes.

The documentation for this class was generated from the following files:

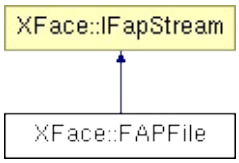
- **FaceBase.h**
- FaceBase.cpp

XFace::FAPFile

XFace::FAPFile Class Reference
[XFace]

```
#include <FapFile.h>
```

Inheritance diagram for XFace::FAPFile:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
June 2003

This class accesses the contents of a FAP file from a disk and using the base class interface (**IFapStream**) provides FAP data to its clients.

Public Member Functions

bool	**isEnd** () const *query open*
bool	**isOpen** () const *query open*
void	**next** ()
void	**rewind** ()
bool	**open** (std::istream &input, const **FAPU** &fapu) *open/read the stream*
virtual size_t	**getFAPCount** () const *Retrieves FAP count.*

Static Public Member Functions

static bool	**save** (**IFapStream** &faps, const **FAPU** &fapu, const std::string &filename) *Saves fap stream to disk.*

Member Function Documentation

void XFace::FAPFile::next ( ) [virtual]

Advances the current FAP to the next one. If we are at the end of the stream, pointer to current FAP is set to 0, so **getCurrentFAP()** call following it will return 0. So this code is legal (althoug it does nothing but goes to the end of the fap stream only);

```
        while(pFAPs->getCurrentFAP())
        {
                pFAPs->next();
        };
```

Implements **XFace::IFapStream**.

void XFace::FAPFile::rewind ( ) [virtual]

Rewinds the stream to the first FAP in the stream.

Implements **XFace::IFapStream**.

bool XFace::FAPFile::save (IFapStream & *faps*, const FAPU & *fapu*, const std::string & *filename*) [static]

Saves a FAP stream to disk.

The documentation for this class was generated from the following files:

- **FapFile.h**
- FapFile.cpp

XFace::FAPU

XFace::FAPU Class Reference
[XFace]

```
#include <FDP.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
June 2003 Used as a data structure for MPEG4 **FAPU** information.

The documentation for this class was generated from the following file:

- **FDP.h**

XFace::FDP

XFace::FDP Class Reference
[XFace]

```
#include <FDP.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
June 2003

MPEG4 **FDP** data is stored in this class.

Public Member Functions

void	**insertItem** (**FDPItem** *item) *Inserts a new **FDPItem** to the list stored inside.*
const **FDPItem** *	**findItem** (const std::string &name, const std::string &aff) const *Finds and retrieves **FDPItem** from the collection stored inside.*
const FDPITEMS &	**getItems** () const *Accessor.*
int	**getItemCount** () const *Accessor.*
void	**setES0** (float es) *Accessor.*
void	**setIRISD0** (float irisd) *Accessor.*
void	**setENS0** (float ens) *Accessor.*
void	**setMNS0** (float mns) *Accessor.*
void	**setMW0** (float mw) *Accessor.*
void	**setFAPU** (const **FAPU** &fapus) *Accessor.*
float	**getES** () const *Accessor.*
float	**getIRISD** () const *Accessor.*
float	**getENS** () const *Accessor.*
float	**getMNS** () const *Accessor.*
float	**getMW** () const *Accessor.*
const **FAPU** &	**getFAPU** () const *Accessor.*
void	**setGlobalTranslation** (const **Vector3** &tr) *Accessor.*
void	**setGlobalTranslation** (float x, float y, float z) *Accessor.*

void	**setGlobalRotation** (const AxisAngle &axisAng) *Accessor.*
void	**setGlobalRotation** (float x, float y, float z, float angle) *Accessor.*
Vector3	**getGlobalRotationAxis** () const *Accessor.*
float	**getGlobalRotationAngle** () const *Accessor.*
const AxisAngle &	**getGlobalAxisAngle** () const *Accessor.*
Vector3	**getGlobalTranslation** () const *Accessor.*

The documentation for this class was generated from the following files:

- **FDP.h**
- FDP.cpp

XFace::FDPFile

XFace::FDPFile Class Reference
[XFace]

```
#include <FDPFile.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Reads/Writes the MPEG-4 Facial Definition Parameters stored in XML format.

The documentation for this class was generated from the following files:

- **FDPFile.h**
- FDPFile.cpp

XFace::FDPItem

XFace::FDPItem Class Reference
[XFace]

```
#include <FDPItem.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
June 2003

MPEG4 **FDP** data is stored in this class. Every **FDPItem** defines a specific **FDP** region on the face model, and all are stored (usually) in **FDP** class as a collection.

Public Member Functions

const AOI_STORE	**getAOI** () const *Accessor.*
const std::string	**getName** () const *Accessor.*
void	**setName** (const std::string &name) *Accessor.*
unsigned short	**getIndex** () const *Accessor.*
bool	**hasControlPoint** () const *Accessor.*
void	**setControlPoint** (unsigned short ind) *Accessor.*
const std::string &	**getAffects** () const *Accessor.*
void	**setAffects** (const std::string &aff) *Accessor.*
void	**addAOIIndex** (unsigned short ind) *Accessor.*
void	**removeAOIIndex** (unsigned short ind) *Accessor.*
bool	**removeInfluenceCalculator** (const std::string &type, float w, unsigned short id) *Accessor.*
void	**modifyInfluenceCalculator** (unsigned int order, const std::string &type, float w, unsigned short fap) *Accessor.*
void	**resetAOI** () *Accessor.*
int	**getAOICount** () const *Accessor.*

The documentation for this class was generated from the following files:

- **FDPItem.h**
- FDPItem.cpp

XFace::FDPLoader

XFace::FDPLoader Class Reference
[XFace]

```
#include <FDPLoader.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
Feb 2005

The documentation for this class was generated from the following files:

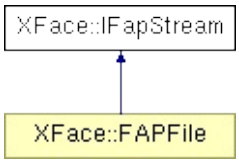
- **FDPLoader.h**
- FDPLoader.cpp

XFace::IFapStream

XFace::IFapStream Class Reference
[XFace]

```
#include <IFapStream.h>
```

Inheritance diagram for XFace::IFapStream:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
June 2003

Interface for the FAP streaming. This base class defines the methodology to access FAP stream. Derived classes may include disk access and network streaming to load the FAPs to this interface.

Public Member Functions

virtual bool	**isEnd** () const =0 *query open*
virtual bool	**isOpen** () const =0 *query open*
virtual bool	**open** (std::istream &, const **FAPU** &)=0 *open/read the stream*
virtual void	**next** ()=0 *Advance to the next FAP.*
virtual void	**rewind** ()=0 *Rewind to the first FAP.*
virtual size_t	**getFAPCount** () const =0 *Retrieves FAP count.*
float	**getVersion** () const *Retrieve file version.*
short	**getFPS** () const *Retrieve number of frames per second.*
const std::vector< float > &	**getCurrentFAP** () const *Retrieve current FAP.*

Protected Attributes

std::vector< float >	**m_currentFAP** *current FAP value*
short	**m_FPS** *Number of frames per second to render.*
float	**m_version** *Version of the file.*

The documentation for this class was generated from the following file:

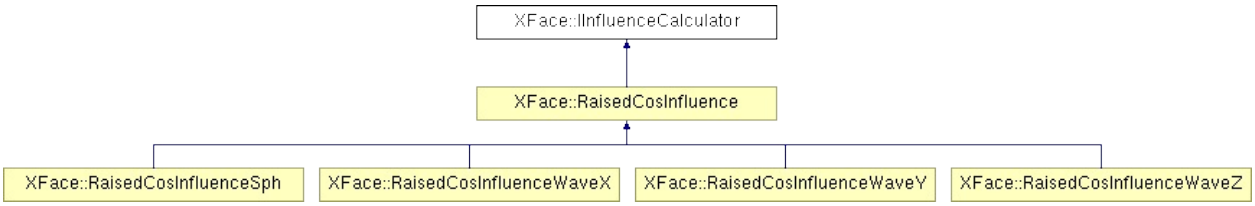
- **IFapStream.h**

XFace::IInfluenceCalculator

XFace::IInfluenceCalculator Class Reference [XFace]

`#include <IInfluenceCalculator.h>`

Inheritance diagram for XFace::IInfluenceCalculator:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
May 2003

IInfluenceCalculator defines the way you should implement new classes to define zone of influence. As this is an abstract base class, you cannot create an instance of **IInfluenceCalculator**, but you should derive your classes from it.

The documentation for this class was generated from the following file:

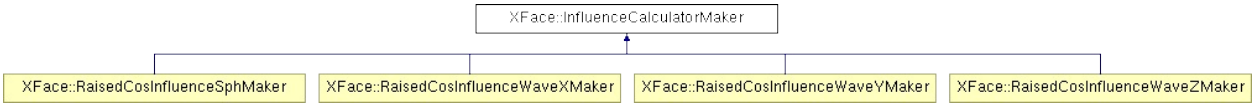
- **IInfluenceCalculator.h**

XFace::InfluenceCalculatorMaker

XFace::InfluenceCalculatorMaker Class Reference
[XFace]

`#include <InfluenceCalculatorMaker.h>`

Inheritance diagram for XFace::InfluenceCalculatorMaker:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
April 2004

Implementation of "Pluggable Factory" design pattern. This class is responsible for creation of **IInfluenceCalculator** derived classes. Note that it is an abstract base class, you should always derive a class and implement **InfluenceCalculatorMaker::makeInfluenceCalculator** function with proper return value specialization for correct **IInfluenceCalculator** type.

An **IInfluenceCalculator** derived class must always have a **InfluenceCalculatorMaker** derived class complementing it. XFaceEd automatically adds available **IInfluenceCalculator** derived classes to its influence set and exposes it to the end user. In addition, only this way the core library can use the influence calculation mechanism properly, for utilizing correct influence calculator.

Static Public Member Functions

static std::vector< std::string >	**getMakerList** ()
static **IInfluenceCalculator** *	**newInfluenceCalculator** (const std::string &, float w, unsigned short fapID)

Protected Member Functions

	InfluenceCalculatorMaker (const std::string &className) *ctor only accessible to derived classes.*
virtual **IInfluenceCalculator** *	**makeInfluenceCalculator** (float w, unsigned short fapID) const =0 *Derived classes should reimplement this pure virtual.*

Constructor & Destructor Documentation

XFace::InfluenceCalculatorMaker::InfluenceCalculatorMaker (cons

Base class constructor registers the derived class identifier to its list of influencers.

Parameters:

className	derived class identifier (should be unique for every influence calc.)

Member Function Documentation

std::vector< std::string > XFace::InfluenceCalculatorMaker::getMak

Returns the list of influence calculator makers as a vector.

Note:
Do NOT call this method frequently, every time a new vector is created and returned.

IInfluenceCalculator * XFace::InfluenceCalculatorMaker::newInflue

Clients should call this static method, to create influence calculators. It accesses the proper factory (maker) and requests a new influence calculator and returns it.

Parameters:

className	derived class unique maker identifier.
weight	with this weight value,
fapID	and used for this FAP.

The documentation for this class was generated from the following files:

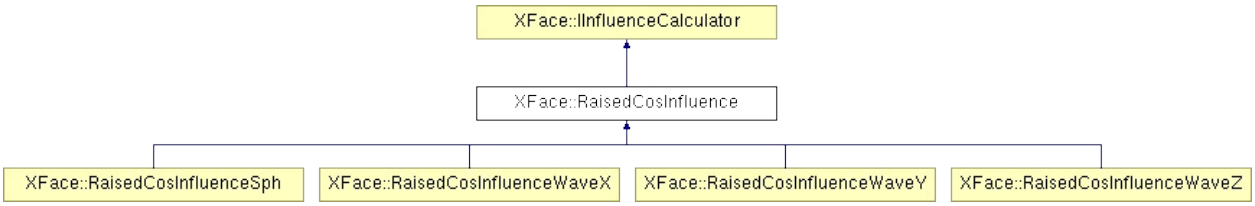
- **InfluenceCalculatorMaker.h**
- InfluenceCalculatorMaker.cpp

XFace::RaisedCosInfluence

XFace::RaisedCosInfluence Class Reference [XFace]

`#include <RaisedCosInfluence.h>`

Inheritance diagram for XFace::RaisedCosInfluence:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
Feb 2004

Implements the base for Raised cosine deformation. Note that, this is still an abstract base class, derive a new class from this one and implement the necessary pure virtual function. You can also use already implemented derived classes for fast use.

The documentation for this class was generated from the following files:

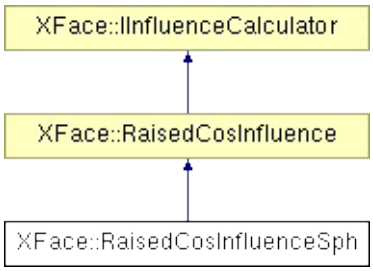
- **RaisedCosInfluence.h**
- RaisedCosInfluence.cpp

XFace::RaisedCosInfluenceSph

XFace::RaisedCosInfluenceSph Class Reference [XFace]

`#include <RaisedCosInfluence.h>`

Inheritance diagram for XFace::RaisedCosInfluenceSph:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
Feb 2004

Spherical deformation is done in this implementation. This means that, distance map to the control point is calculated in 3 dimensions, and when deformation is applied, every vertex affected moves according to its 3d distance to the control point.

The documentation for this class was generated from the following files:

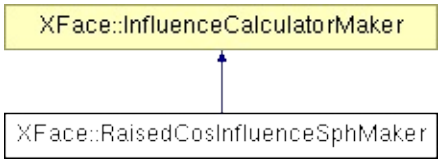
- **RaisedCosInfluence.h**
- RaisedCosInfluence.cpp

XFace::RaisedCosInfluenceSphMaker

XFace::RaisedCosInfluenceSphMaker Class Reference
[XFace]

`#include <RaisedCosInfluence.h>`

Inheritance diagram for XFace::RaisedCosInfluenceSphMaker:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
Feb 2004

Maker for spherical deformation.

The documentation for this class was generated from the following files:

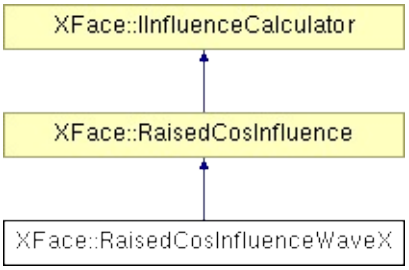
- **RaisedCosInfluence.h**
- RaisedCosInfluence.cpp

XFace::RaisedCosInfluenceWaveX

XFace::RaisedCosInfluenceWaveX Class Reference
[XFace]

```
#include <RaisedCosInfluence.h>
```

Inheritance diagram for XFace::RaisedCosInfluenceWaveX:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
Feb 2004

Wave deformation is done in this implementation. This means that, distance map to the control point is calculated in only one particular dimensions (here in X), and when deformation is applied, every vertex affected moves according to that distance to the control point.

The documentation for this class was generated from the following files:

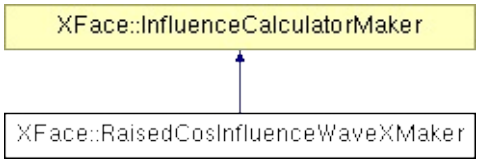
- **RaisedCosInfluence.h**
- RaisedCosInfluence.cpp

XFace::RaisedCosInfluenceWaveXMaker

XFace::RaisedCosInfluenceWaveXMaker Class Reference
[XFace]

```
#include <RaisedCosInfluence.h>
```

Inheritance diagram for XFace::RaisedCosInfluenceWaveXMaker:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
Feb 2004

Maker for wave deformation in X direction.

The documentation for this class was generated from the following files:

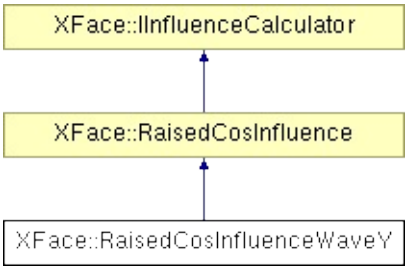
- **RaisedCosInfluence.h**
- RaisedCosInfluence.cpp

XFace::RaisedCosInfluenceWaveY

XFace::RaisedCosInfluenceWaveY Class Reference
[XFace]

```
#include <RaisedCosInfluence.h>
```

Inheritance diagram for XFace::RaisedCosInfluenceWaveY:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
Feb 2004

Wave deformation is done in this implementation. This means that, distance map to the control point is calculated in only one particular dimensions (here in Y), and when deformation is applied, every vertex affected moves according to that distance to the control point.

The documentation for this class was generated from the following files:

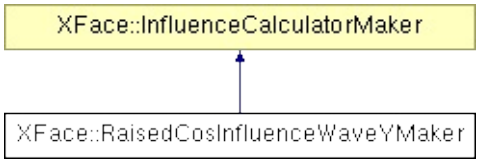
- **RaisedCosInfluence.h**
- RaisedCosInfluence.cpp

XFace::RaisedCosInfluenceWaveYMaker

XFace::RaisedCosInfluenceWaveYMaker Class Reference
[XFace]

`#include <RaisedCosInfluence.h>`

Inheritance diagram for XFace::RaisedCosInfluenceWaveYMaker:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
Feb 2004

Maker for wave deformation in Y direction.

The documentation for this class was generated from the following files:

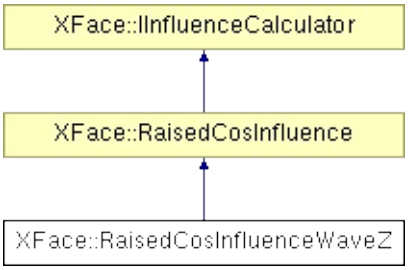
- **RaisedCosInfluence.h**
- RaisedCosInfluence.cpp

XFace::RaisedCosInfluenceWaveZ

XFace::RaisedCosInfluenceWaveZ Class Reference
[XFace]

`#include <RaisedCosInfluence.h>`

Inheritance diagram for XFace::RaisedCosInfluenceWaveZ:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
Feb 2004

Wave deformation is done in this implementation. This means that, distance map to the control point is calculated in only one particular dimensions (here in Z), and when deformation is applied, every vertex affected moves according to that distance to the control point.

The documentation for this class was generated from the following files:

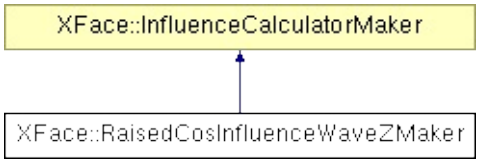
- **RaisedCosInfluence.h**
- RaisedCosInfluence.cpp

XFace::RaisedCosInfluenceWaveZMaker

XFace::RaisedCosInfluenceWaveZMaker Class Reference
[XFace]

`#include <RaisedCosInfluence.h>`

Inheritance diagram for XFace::RaisedCosInfluenceWaveZMaker:



List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
Feb 2004

Maker for wave deformation in Z direction.

The documentation for this class was generated from the following files:

- **RaisedCosInfluence.h**
- RaisedCosInfluence.cpp

XFace::XercesString

XFace::XercesString Class Reference
[XFace]

```
#include <XercesString.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
Feb 2005

Handles Xerces string operations, taken from sample in http://www-106.ibm.com/developerworks/xml/library/x-xercc2/ written by Rick Parrish

Public Member Functions

bool	**operator!=** (const char *str) const

Member Function Documentation

bool XFace::XercesString::operator!= (const char * *str*) const

added by Koray Balci, Feb 2005

The documentation for this class was generated from the following files:

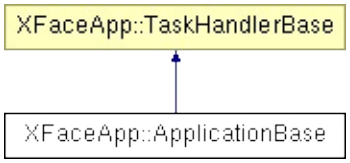
- **XercesString.h**
- XercesString.cpp

XFaceApp::ApplicationBase

XFaceApp::ApplicationBase Class Reference

`#include <ApplicationBase.h>`

Inheritance diagram for XFaceApp::ApplicationBase:



List of all members.

Detailed Description

Author:
Koray Balci

Abstract base class for Xface related **Task** processing. Implements the empty **Task** handlers defined in base class **TaskHandlerBase**, but pure virtual create* methods also enforces derivation from this class. For basic use, task handlers in this class does the job, at least Xface related job. For example, for adding GUI related things, you should reimplement these task handlers, and call the implementations provided here from derived ones. So, the main idea is to provide the basic mechanism and hide the Xface related implementation while letting users to develop their own application using various available libraries and technologies.

Public Types

enum	**QUERY**

Public Member Functions

virtual void	**muteAudio** (bool bSnd)
bool	**query** (const **QUERY** &q)
virtual bool	**init** ()

Protected Member Functions

virtual bool	**yield** ()=0
virtual bool	**createScriptProcessors** ()=0 *Creates an object implementing IScriptProcessor interface.*
virtual boost::shared_ptr< **ITimer** >	**createTimer** () const =0 *Creates an object implementing ITimer interface.*
virtual boost::shared_ptr< **ISound** >	**createSound** () const =0 *Creates an object implementing **ISound** interface.*
virtual boost::shared_ptr< **FaceBase** >	**createFace** () const =0 *Creates a FaceBase object (or a derived one).*
virtual boost::shared_ptr< **IRenderer** >	**createRenderer** () const =0 *Creates an object implementing XFace::IRenderer interface.*
virtual boost::shared_ptr< **IFapStream** >	**createFapStream** () const =0 *Creates an object implementing IFapStream interface.*
virtual void	**fireNotification** (const **Notification** &note) const =0
virtual void	**renderBegin** () const
virtual void	**renderEnd** () const
unsigned long	**synchronize** (bool)
virtual void	**onRenderFrame** () const ***Task** Handler.*
virtual void	**onAdvanceFrame** ()
virtual bool	**onResumePlayback** ()
virtual void	**onStopPlayback** ()
virtual void	**onRewindPlayback** () const
virtual bool	**onLoadFDP** (const std::string &param1, const std::string &param2)
virtual bool	**onLoadFAP** (const std::string &param)
virtual bool	**onLoadPHO** (const std::string &param1, const std::string &param2)
virtual bool	**onLoadANIM** (const std::string &param1)
virtual bool	**onLoadWAV** (const std::string &param)
virtual bool	**onUpLoadFAP** (const std::string &param)
virtual bool	**onUpLoadPHO** (const std::string &param1, const std::string &param2, const std::string &param3)
virtual bool	**onUpLoadANIM** (const std::string &param1)

virtual bool	**onUpLoadScript** (const std::string &param1, const std::string &param2)

Member Enumeration Documentation

enum XFaceApp::ApplicationBase::QUERY

Basic query mechanism. To be changed..

Member Function Documentation

virtual void XFaceApp::ApplicationBase::fireNotification (const No

Fires the **Notification**, implementation is probably TCP/IP related, hence left out of core library and implemented in derived class in XfacePlayer.

bool XFaceApp::ApplicationBase::init ( ) [virtual]

Pure virtual create* methods are called here to create necessary objects and stored in interface pointers. This way, we provide the mechanism but developers might be interested/need using platform dependent or external library dependent classes, or just improve the already supplied classes in Xface.

Sound, Xface::FaceBase, timer, renderer, fap stream, script processor objects are created.

void XFaceApp::ApplicationBase::muteAudio (bool *bSnd*) [virtu

Mutes the audio. Good for testing purposes during development!

void XFaceApp::ApplicationBase::onAdvanceFrame ( ) [protected

Advances the FAP frame, and updates the deformation. No **Notification** is fired.

bool XFaceApp::ApplicationBase::onLoadANIM (const std::string

Loads the ANIM file from the server (XfacePlayer) side.

Parameters:

param1	is the filename **Notification::kError** is fired on error and **Notification::kFinished** is fired on success.

Reimplemented from **XFaceApp::TaskHandlerBase**.

bool XFaceApp::ApplicationBase::onLoadFAP (const std::string &

Loads the FAP file from the server (XfacePlayer) side.

Parameters:

param is the filename **Notification::kError** is fired on error and **Notification::kFinished** is fired on success.

Reimplemented from **XFaceApp::TaskHandlerBase**.

bool XFaceApp::ApplicationBase::onLoadFDP (const std::string &
const std::string &
) [protected, virtua

Loads the FDP file from the server (XfacePlayer) harddisk.

Parameters:

param1 is the filename

param2 is the path **Notification::kError** is fired on error and **Notification::kFinished** is fired on success.

Reimplemented from **XFaceApp::TaskHandlerBase**.

bool XFaceApp::ApplicationBase::onLoadPHO (const std::string &
const std::string &
) [protected, virtua

Loads the PHO file from the server (XfacePlayer) side.

Parameters:

param1 is the filename

param2 is the language **Notification::kError** is fired on error and **Notification::kFinished** is fired on success.

Reimplemented from **XFaceApp::TaskHandlerBase**.

bool XFaceApp::ApplicationBase::onLoadWAV (const std::string &

Loads the WAV file from the server (XfacePlayer) disk.

Parameters:

param is the filename **Notification::kError** is fired on error and **Notification::kFinished** is fired on success.

Reimplemented from **XFaceApp::TaskHandlerBase**.

void XFaceApp::ApplicationBase::onRenderFrame ( ) const [prot

Renders the current frame. No **Notification** is fired.

Reimplemented from **XFaceApp::TaskHandlerBase**.

bool XFaceApp::ApplicationBase::onResumePlayback ( ) [protec

Starts/Resumes the playback of FAP stream and the WAV file and sends back **Notification::kStarted** in the beginning and **Notification::kFinished** in the end to the client. On error, **Notification::kError** is returned. Also yields the message pump for checking if there is a pause or stop **Task** arrived or not. If by chance, another resume playback **Task** is in the task queue, it is not executed until tht current task finishes.

Reimplemented from **XFaceApp::TaskHandlerBase**.

void XFaceApp::ApplicationBase::onRewindPlayback ( ) const [p

Rewinds the FAP stream and the WAV file to beginning and fires **Notification::kFinished**.

Reimplemented from **XFaceApp::TaskHandlerBase**.

void XFaceApp::ApplicationBase::onStopPlayback ( ) [protected,

Stops and rewinds the FAP stream and the WAV file to beginning and fires **Notification::kFinished**.

Reimplemented from **XFaceApp::TaskHandlerBase**.

bool XFaceApp::ApplicationBase::onUpLoadANIM (const std::stri

Parameters:

param1 is the ANIM contents

Reimplemented from **XFaceApp::TaskHandlerBase**.

bool XFaceApp::ApplicationBase::onUpLoadFAP (const std::strin

Parameters:

param is the FAP contents

Reimplemented from **XFaceApp::TaskHandlerBase**.

bool XFaceApp::ApplicationBase::onUpLoadPHO (const std::strin
const std::strin
const std::strin
) [protected, vir

Parameters:

param1 is the PHO contents

param2 is the PHO language

Reimplemented from **XFaceApp::TaskHandlerBase**.

bool XFaceApp::ApplicationBase::onUpLoadScript (const std::str
const std::str
) [protected, vi

Processes the script string passed (from remote client possibly). Current IScriptProcessor implementation (Expl2fapProcessor class) also loads the FAP and WAV file produced automatically by issuing proper Tasks.

Parameters:

param1	is the script string to process
param2	is the name of script processor **Notification::kError** is sent on error and **Notification::kStarted** is sent on start and **Notification::kFinished** when finished, to the client.

Reimplemented from **XFaceApp::TaskHandlerBase**.

bool XFaceApp::ApplicationBase::query (const QUERY & *q*)

Provides basic query mechanism.

Todo:

The mechanism can be improved and reverted to Task/Notification type easily. Or at least we can add a new **Task** type for QUERY and implement handlers.

virtual void XFaceApp::ApplicationBase::renderBegin ( ) const [:

Some GUI libraries (wxWidgets, SDL, MFC) for OpenGL windowing operation require pre and especially post processing for double buffering. Derived classes should re-implement these empty methods properly

virtual void XFaceApp::ApplicationBase::renderEnd ( ) const [in

Some GUI libraries (wxWidgets, SDL, MFC) for OpenGL windowing operation require pre and especially post processing for double buffering. Derived classes should re-implement these empty methods properly

unsigned long XFaceApp::ApplicationBase::synchronize (bool *bS*

Synchronizes the animation to frames per seconds required by current FAP being played back.

Todo:

Check this function, not tested extensively. Especially in debug mode, we seem to have some problem, should skip the frames, but not as it seems, lagging animation.

virtual bool XFaceApp::ApplicationBase::yield ( ) [protected, pur

You should provide an application specific (probably external library dependent) yield functionality which enables processing the task queue, while a task is already being done. We use wxWidgets yield method to achieve this in XfacePlayer implementation (see wxFace class).

The documentation for this class was generated from the following files:

- **ApplicationBase.h**
- ApplicationBase.cpp

XFaceApp::ISound

XFaceApp::ISound Class Reference

```
#include <ISound.h>
```

List of all members.

Detailed Description

Author:

Koray Balci Provides the interface for sound playback operation.

The documentation for this class was generated from the following file:

- **ISound.h**

XFaceApp::ModelCamera

XFaceApp::ModelCamera Class Reference

```
#include <ModelCamera.h>
```

List of all members.

Detailed Description

Author:

Koray Balci

Todo:

This class should be in another name space, in some other place, **XEngine** perhaps?

The documentation for this class was generated from the following files:

- **ModelCamera.h**
- ModelCamera.cpp

XFaceApp::Notification

XFaceApp::Notification Class Reference

`#include <Notification.h>`

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
June 2004

Notification objects are used for communication between the client and the server. Server always sends clients messages as **Notification** objects translated to XML strings. Every **Notification** object represents the status of a **Task** object (only ERROR notification is a small exception to that). So, owner and task ID properties are the same as of **Task** object's.

Public Types

enum	**TaskStatus** {
	kUnknown = 0, **kQueued**, **kStarted**, **kFinished**,
	kDiscarded, **kInProgress**, **kInQueue**, **kError**
	}

Public Member Functions

	Notification (const std::string &name, unsigned short owner=0, unsigned int task=0) *Creates a **Notification** with default values for the given name as parameter.*
	Notification (const **Task** &task, const **TaskStatus** &status) *Creates **Notification** for the given **Task**.*

void	**setStatus** (**TaskStatus** status) *Accessor.*
TaskStatus	**getStatus** () const *Accessor.*
unsigned short	**getOwnerID** () const *Accessor.*
unsigned long	**getTaskID** () const *Accessor.*
const std::string &	**getName** () const *Accessor.*

Static Public Member Functions

static **Notification**	**error** () *In case of unknown errors, we use this canned method that creates an error notification.*

Member Enumeration Documentation

enum XFaceApp::Notification::TaskStatus

Status/Fate of the task can be one of these.

Enumerator:

kUnknown	Who knows what happened to the poor task, used for initialization only/usually.
kQueued	Acknowledgement notification sent immediately after the **Task** is received & queued.
kStarted	**Task** is started, notify the **Task** owner.
kFinished	**Task** is finished, notify the **Task** owner.
kDiscarded	**Task** is discarded (very unprobable) by some overriding task.
kInProgress	Client querried the task, and it is actually in progress at the moment.
kInQueue	Client querried the task, and it is still in task queue.
kError	Something went wrong somewhere.

The documentation for this class was generated from the following files:

- **Notification.h**
- Notification.cpp

XFaceApp::Task

XFaceApp::Task Class Reference

`#include <Task.h>`

List of all members.

Detailed Description

Author:
Koray Balci

For controlling Xface, clients have one of two ways. First is the use of classes implemented as usual, and the second is using the **Task** mechanism. We define a set of **Task** types, and this class stores the properties of these tasks. Every **Task** object has a name, set of parameters, an ID and owner ID. Name is a string representing the actual role of the object. You can find more information about **Task** names/types in **TaskDictionary** class documentation. Every **Task** object has also a set of string parameters depending on the **Task** type it belongs to. ID is supposed to be a unique number representing the **Task** instance. It might have been assigned automatically during construction, but we chose it to be assigned externally for the time being. Owner ID represents the thread/application which has created and issued this task to Xface. This value is used for feedback notification (more on this: **Notification** class) mechanism.

Here is a sample of usage from XfaceClient, where we issue a "resume playback" task through TCP/IP;

```
        // we create a task with the client Id and a unique task ID.
        Task task("RESUME_PLAYBACK", m_clientId, ++m_taskCount + 100000
        // following function translates the Task to XML string
        std::string msg = XMLUtils::xmlify(task);
        // Here we send the string through network
        m_pSocket->Write(msg.c_str(), msg.size() + 1);
```

On the other hand, if this is a standalone application, or we have direct access to **XFaceApp::TaskHandlerBase** derived class, we can issue the **Task** as follows;

```
        Task playtask("RESUME_PLAYBACK");
        m_pApp->newTask(playtask);      // m_pApp is a pointer to a Tas
```

Note that, we do not have to (and in fact should not) pass owner id and task id for the **Task** object, because, it is defaulted to 0, and this way we prevent creating/sending **Notification** messages. After all, the latter use

is only for the thread that has access to the **TaskHandlerBase** class. A class derived from **TaskHandlerBase**, tailored for your own purposes can implement event notification mechanism much more efficiently.

Future: In the future versions, we can add plenty of things to this mechanism. Parameters can be in types other than string. ID can be assigned automatically and invisibly, however this can create a set of new problems, not sure.. And finally, we can implement a whole class hierarchy instead of using strings for naming **Task** types. For the time being, the system is sufficient for our needs.

Public Member Functions

void	**clearParams** () *Clears the parameters.*
unsigned short	**getParamCount** () const *Returns the number of currently pushed parameters.*
void	**setName** (const std::string &name) *Sets the name, first checks the existence of the **Task** by calling **TaskDictionary::isTask**.*
const std::string &	**getName** () const *Returns the name of the **Task**.*
void	**pushParameter** (const std::string &param) *Pushes a new parameter string to the parameter collection.*
std::string	**getParameter** (unsigned int id) const *Returns the parameter indexed by id.*
unsigned short	**getOwnerID** () const *Returns the owner ID.*
void	**setOwnerID** (unsigned short id=0) *Sets the owner ID.*
unsigned int	**getID** () const *Gets **Task** ID.*
void	**setID** (unsigned int id=0) *Sets **Task** ID.*
	Task (const std::string &name, unsigned short owner=0, unsigned int id=0)

Constructor & Destructor Documentation

```
XFaceApp::Task::Task ( const std::string & name,
                       unsigned short      owner = 0,
                       unsigned int        id = 0
                     ) [inline]
```

Note that during construction, task name is checked against **TaskDictionary**, so, if an invalid task name is inserted, the name of the task is empty string. If you are not sure about the validity of the task name, try calling **TaskDictionary::isTask** beforehand.

Member Function Documentation

std::string XFaceApp::Task::getParameter (unsigned int *id*) cons

Gets the parameter indexed at id. If an invalid (out of bounds) index is passed an empty string is returned.

The documentation for this class was generated from the following files:

- **Task.h**
- Task.cpp

XFaceApp::TaskDictionary

XFaceApp::TaskDictionary Class Reference

`#include <Task.h>`

List of all members.

Detailed Description

Author:
Koray Balci

TaskDictionary class stores the list of **Task** types, and serve other classes as a dictionary to check. You can query the existance of a **Task** type, get the number of parameters, and also add new **Task** types on the fly to extend the capabilities of the library. Note that, **TaskDictionary** is a singleton and a monolithic class with all member functions being static, therefore you can only have one dictionary of tasks per application instance.

Here is the list of currently available **Task** types and their meanings;

- **RESUME_PLAYBACK** Starts the playback of the currently loaded FAP file.
  See also:
  ApplicationBase::onResumePlayback
- **STOP_PLAYBACK**
  Stops the playback and rewinds the FAP file to the beginning.
  See also:
  ApplicationBase::onStopPlayback
- **PAUSE_PLAYBACK**
  Pauses the playback.
  See also:
  ApplicationBase::onPausePlayback
- **REWIND_PLAYBACK**
  Rewinds the playback.
  See also:
  ApplicationBase::onRewindPlayback
- **RENDER_FRAME**
  Renders the current frame.
  See also:
  ApplicationBase::onRenderFrame
- **ADVANCE_FRAME**
  Advances the frame
  See also:

ApplicationBase::onAdvanceFrame

- **LOAD_FAP**
  Loads FAP
  See also:
  ApplicationBase::onLoadFAP
- **LOAD_WAV**
  Loads WAV
  See also:
  ApplicationBase::onLoadWAV
- **LOAD_FDP**
  Loads FDP
  See also:
  ApplicationBase::onLoadFDP
- **LOAD_PHO**
  Loads PHO
  See also:
  ApplicationBase::onLoadPHO
- **UPLOAD_FAP**
  UpLoads FAP
  See also:
  ApplicationBase::onUpLoadFAP
- **UPLOAD_PHO**
  UpLoads PHO
  See also:
  ApplicationBase::onUpLoadPHO
- **UPLOAD_SCRIPT**
  Uploads script
  See also:
  ApplicationBase::onUpLoadScript
- **UPLOAD_WAV**
  Used currently in a bad way, send filename and size of WAV file as parameter, and then after sending the **Task**, send the WAV file in binary from the same communication port. Specific to XfacePlayer only for the moment.
  Todo:
  Should implement a better way to upload binary files first to implement this (UPLOAD_WAV) **Task**, note that it is also necessary to be able to upload binary (compressed) FAP files.
  See also:

ApplicationBase::onUpLoadANIM

- **LOAD_ANIM**
  Loads ANIM file created by SMIL-Agent (temporary addition)
  See also:
  ApplicationBase::onLoadANIM

Static Public Member Functions

static bool	**isTask** (const std::string &name)
static int	**queryParamCount** (const std::string &name)
static void	**registerTask** (const std::string &name, int nParam)

Member Function Documentation

static bool XFaceApp::TaskDictionary::isTask (const std::string &

Parameters:

name **Task** name

Returns:

true if it is a **Task**

static int XFaceApp::TaskDictionary::queryParamCount (const std

Parameters:

name **Task** name

Returns:

number of parameters

static void XFaceApp::TaskDictionary::registerTask (const std::str int) [inline, stat

Used to register a new **Task** type.

Parameters:

name **Task** name

nParam Number of parameters

The documentation for this class was generated from the following files:

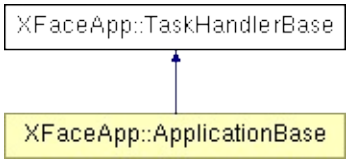
- **Task.h**
- Task.cpp

XFaceApp::TaskHandlerBase

XFaceApp::TaskHandlerBase Class Reference

`#include <TaskHandlerBase.h>`

Inheritance diagram for XFaceApp::TaskHandlerBase:



List of all members.

Detailed Description

Author:
Koray Balci

Task objects are issued and processed using this class. Current implementation has a **Task** queue (std::queue), and you can push new Tasks to the queue using **TaskHandlerBase::newTask** method. Then, when **TaskHandlerBase::processTask** method is called, the **Task** in queue is popped, and processed using TaskHandlerBase::dispatch method. In TaskHandlerBase::dispatch (private & non-virtual method), depending on **Task** type, the task is routed to appropriate handler method. If a time spanning task is being processed when the system tries to process a new task, it automatically waits until the end of the current task. That's why time spanning task handler methods return boolean values. All the task handler methods are defined as virtual with empty bodies, which means this class, by itself does nothing. You should never use this class directly, hence the destructor of the class is protected. This class only implements and hides the **Task** processing mechanism from its clients.

See also:
ApplicationBase

Public Member Functions

virtual void	**newTask** (const **Task** &task) *Post a new **Task** through here.*
virtual bool	**processTask** () *Process the **Task** queue.*
void	**discardPendingTasks** () *Discard all pending **Task** objects in the queue.*
void	**pause** (bool bPause=true) *Pauses the system.*

Protected Member Functions

virtual void	**onRenderFrame** () const ***Task*** *handler with empty body.*
virtual void	**onAdvanceFrame** () const ***Task*** *handler with empty body.*
virtual bool	**onResumePlayback** () ***Task*** *handler with empty body.*
virtual void	**onStopPlayback** () ***Task*** *handler with empty body.*
virtual void	**onRewindPlayback** () const ***Task*** *handler with empty body.*
virtual bool	**onLoadFDP** (const std::string &, const std::string &) ***Task*** *handler with empty body.*
virtual bool	**onLoadFAP** (const std::string &) ***Task*** *handler with empty body.*
virtual bool	**onLoadPHO** (const std::string &, const std::string &) ***Task*** *handler with empty body.*
virtual bool	**onLoadANIM** (const std::string &) ***Task*** *handler with empty body.*
virtual bool	**onLoadWAV** (const std::string &) ***Task*** *handler with empty body.*
virtual bool	**onUpLoadFAP** (const std::string &) ***Task*** *handler with empty body.*
virtual bool	**onUpLoadPHO** (const std::string &, const std::string &, const std::string &) ***Task*** *handler with empty body.*
virtual bool	**onUpLoadANIM** (const std::string &) ***Task*** *handler with empty body.*
virtual bool	**onUpLoadScript** (const std::string &, const std::string &) ***Task*** *handler with empty body.*

Protected Attributes

Task	**m_currentTask**
	Pointer to current task that is being processed is stored for firing proper Notification(s).

Member Function Documentation

void XFaceApp::TaskHandlerBase::discardPendingTasks ( )

Empties all the tasks in the queue. Not to be abused, might be necessary for high priority tasks like saving avi, and can be called before issueing such tasks.

void XFaceApp::TaskHandlerBase::newTask (const Task & *task*)

Pushes the **Task** directly to the task queue.

bool XFaceApp::TaskHandlerBase::processTask ( ) [virtual]

Pops the front task from the queue and tries to dispatch. Should be called externally every now and then (even when a task is being processed), derived class **ApplicationBase** does it well.

Returns:

false only if the queue is empty, true otherwise. Even if the task could not be dispatched, still we return true, because it means the system is busy, so we put back the task in the queue actually.

See also:

TaskHandlerBase::dispatch

The documentation for this class was generated from the following files:

- **TaskHandlerBase.h**
- TaskHandlerBase.cpp

XFaceApp::XMLUtils

XFaceApp::XMLUtils Class Reference

```
#include <XMLUtils.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Various utility functions for XML go here. All members are static, this class exists just to avoid polluting global namespace.

17.02.2005: Added Xerces library and removed TinyXML use.

Static Public Member Functions

static std::string	**xmlify** (const **Task** &task) *Creates the XML sentence for the **Task** task as a std::string.*
static std::string	**xmlify** (const **Notification** &note) *Creates the XML sentence for the **Notification** note as a std::string.*
static **Notification**	**xmlToNotification** (const std::string &xmlString) *Creates the **Notification** object from the XML sentence passed as a std::string.*
static **Task**	**xmlToTask** (const std::string &xmlString) *Creates the **Task** object from the XML sentence passed as a std::string.*

The documentation for this class was generated from the following files:

- **XMLUtils.h**
- XMLUtils.cpp

XMath::LineSegment3

XMath::LineSegment3 Class Reference
[XMath]

```
#include <XLineSegment.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
April 2003

Class that represents a line segment in 3D, no operations (methods) are involved, only data abstraction for the time being.

Public Member Functions

const **Vector3** &	**getStart** () const *Accessor.*
const **Vector3** &	**getEnd** () const *Accessor.*
void	**setStart** (const **Vector3** &s) *Accessor.*
void	**setEnd** (const **Vector3** &e) *Accessor.*

The documentation for this class was generated from the following file:

- **XLineSegment.h**

XMath::Matrix4

XMath::Matrix4 Class Reference
[XMath]

```
#include <Matrix4.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0 (April 2002) 1.1

Date:
April 2002

Class which defines 4x4 matrix and related/necessary mathematical operations.

Matrix data are stored in a one dimensional array in OpenGL format. Check the additional constructor and () operator overload implementation for accessing data as if it is a two dimensional array.

Note:
No inlining has been done yet. All the inlines and other optimizations are to be added in a later stage when we have a better understanding of bottlenecks.

Public Member Functions

void	**negate** (void) *Negates.*
Matrix4	**getInverseTransform** () const *Switches to inverse of this matrix.*
void	**loadIdentity** () *Switches to Identity matrix.*
Matrix4	**getTranspose** () const *returns Transpose*
	operator float * () *float* cast overloaded*
	operator const float * () const *float* cast overloaded*
Matrix4	**operator *** (const **Matrix4** &rhs) ** operator overloaded (matrix)*
Matrix4d	**operator *** (const Matrix4d &rhs) ** operator overloaded (matrix double)*
Matrix4	**operator *** (const float scalar) const ** operator overloaded (scalar)*
float &	**operator()** (unsigned int i, unsigned int j) *() operator overloaded*
float &	**operator[]** (int i) *[] operator overloaded*
float	**operator[]** (int i) const *[] operator overloaded (const version)*
Matrix4 &	**operator=** (const **Matrix4** &rhs) *= operator overloaded*
Vector3	**operator *** (const **Vector3** &rhs) const ** operator overloaded (vector)*
const float *	**getMatrix** () *Accessor function for the data member.*
	Matrix4 (const **Matrix4** &pCopy) *Copy Constructor.*
	Matrix4 () *Default Constructor.*
	Matrix4 (float f00, float f01, float f02, float f03, float f10, float f11, float f12, float f13, float f20, float f21, float f22, float f23, float f30, float f31, float f32, float f33) *Additional constructor for easy initialization.*

	Matrix4 (const float mat[16]) *Additional constructor for easy initialization.*
	~Matrix4 () *Destructor.*

Member Function Documentation

Matrix4 XMath::Matrix4::getInverseTransform ( ) const

Gets the inverse of the transformation matrix.

Note:
Only usable for OpenGL transform matrix inverse

Matrix4 XMath::Matrix4::getTranspose ( ) const

Returns the transpose of the current matrix data. Does not change the content of this matrix.

void XMath::Matrix4::loadIdentity ( )

Loads the identity matrix, erasing current data.

void XMath::Matrix4::negate (void)

Negates the contents of the matrix.

Vector3 XMath::Matrix4::operator * (const Vector3 & *rhs*) const

Matrix by Vector Multiplication operator overloaded.

Warning:
Multiplied according to OpenGL matrix format!

Matrix4 XMath::Matrix4::operator * (const float *scalar*) const

Matrix by Scalar Multiplication operator overloaded.

Note:
Not tested and can be optimized as well.

Matrix4d XMath::Matrix4::operator * (const Matrix4d & *rhs*)

Matrix by Matrix Multiplication operator overloaded.

Warning:
Not tested and can be optimized as well.

Matrix4 XMath::Matrix4::operator * (const Matrix4 & *rhs*)

Matrix by Matrix Multiplication operator overloaded.

Warning:
Not tested and can be optimized as well.

The documentation for this class was generated from the following files:

- Matrix4.h
- Matrix4.cpp

XMath::Quaternion

XMath::Quaternion Class Reference

`#include <Quaternion.h>`

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
May 2002

Quaternion class for robust 3D operations (rotation especially).

Note:
No inlining has been done yet. All the inlines and other optimizations are to be added in a later stage when we have a better understanding of bottlenecks.

Public Member Functions

Quaternion &	**FromRotationMatrix** (**Matrix4** &mat)
Matrix4	**ToRotationMatrix** () const
void	**Identity** () *Loads identity quaternion.*
	Quaternion (float pitch, float yaw, float roll) *Additional Constructor.*
	Quaternion (float ix, float iy, float iz, float iw) *Additional Constructor.*
	Quaternion (const AxisAngle &axisAngle) *Additional Constructor.*
	Quaternion (const **Quaternion** &rhs) *Copy Constructor.*
	Quaternion () *Default Constructor.*
Quaternion &	**operator=** (const **Quaternion** &rhs) *= operator overloaded*
Quaternion	**operator *** (const **Quaternion** &rhs) const ** operator overloaded*
Quaternion	**operator+** (const **Quaternion** &rhs) const *+ operator overloaded*

Public Attributes

float	**x** *Data members.*

Constructor & Destructor Documentation

XMath::Quaternion::Quaternion (float *pitch*,
float *yaw*,
float *roll*
)

Constructor for creating quaternion from euler angles

Parameters:

pitch angle for X axis in degrees
yaw angle for Y axis in degrees
roll angle for Z axis in degrees

XMath::Quaternion::Quaternion (const AxisAngle & *axisAngle*)

Constructor for creating quaternion from axis angle representation

Parameters:

axisAngle axis-angle representation

Member Function Documentation

Quaternion & XMath::Quaternion::FromRotationMatrix (Matrix4 &

Passage between matrix->quaternion representation

Parameters:

mat A valid 4x4 rotation matrix

Matrix4 XMath::Quaternion::ToRotationMatrix ( ) const

Passage between quaternion->matrix representation.

Returns:

A valid 4x4 rotation matrix.

The documentation for this class was generated from the following files:

- Quaternion.h
- Quaternion.cpp

XMath::Ray3

XMath::Ray3 Class Reference
[XMath]

```
#include <Ray3.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
April 2003

Class that represents a ray in 3D, no operations (methods) are involved, only data abstraction for the time being.

Public Member Functions

const **Vector3** &	**getOrigin** () const *Accessor.*
const **Vector3** &	**getDirection** () const *Accessor.*
void	**setOrigin** (const **Vector3** &org) *Accessor.*
void	**setDirection** (const **Vector3** &dir) *Accessor.*

The documentation for this class was generated from the following file:

- **Ray3.h**

XMath::Rectangle2

XMath::Rectangle2 Class Reference
[XMath]

```
#include <Rectangle.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
April 2003

Class that represents a rectangle in 2D, no operations (methods) are involved, only data abstraction for the time being.

Public Member Functions

bool	**isPointInRect** (float x, float y) const
const **Vertex2D** &	**getTopLeft** () const *Accessor.*
const **Vertex2D** &	**getBottomRight** () const *Accessor.*
float	**getX0** () const *Accessor.*
float	**getY0** () const *Accessor.*
float	**getX1** () const *Accessor.*
float	**getY1** () const *Accessor.*
float	**getWidth** () const *Accessor.*
float	**getHeight** () const *Accessor.*
void	**setX0** (float _x0) *Accessor.*
void	**setY0** (float _y0) *Accessor.*
void	**setX1** (float _x1) *Accessor.*
void	**setY1** (float _y1) *Accessor.*

Member Function Documentation

bool XMath::Rectangle2::isPointInRect (float *x*, float *y*) const `[inline]`

Checks whether the x/y values lie inside the rectangle.

Note:
top bottom x/y reversed!! it's not a bug but just convention..

The documentation for this class was generated from the following file:

- **Rectangle.h**

XMath::Triangle3

XMath::Triangle3 Class Reference
[XMath]

```
#include <XTriangle.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
April 2003

Class that represents a triangle in 3D, no operations (methods) are involved, only data abstraction for the time being.

Public Member Functions

	Triangle3 (const **Vector3** &p0, const **Vector3** &p1, const **Vector3** &p2) *Creates a triangle from three vectors (vectors represent 3d points!! not edges).*
const **Vector3** &	**getOrigin** () const *Accessor.*
const **Vector3** &	**getEdge0** () const *Accessor.*
const **Vector3** &	**getEdge1** () const *Accessor.*
void	**setOrigin** (const **Vector3** &org) *Accessor.*
void	**setEdge0** (const **Vector3** &ed0) *Accessor.*
void	**setEdge1** (const **Vector3** &ed1) *Accessor.*

The documentation for this class was generated from the following file:

- **XTriangle.h**

XMath::Vector3

XMath::Vector3 Class Reference
[XMath]

```
#include <Vector3.h>
```

List of all members.

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
April 2002

Class that represents a vector in 3D, and necessary mathematical operations.

Note:
A small bit of advice can be usage of *=, +=, -= instead of *, +, - for optimization purposes. Because, each time the latter versions are called, a new **Vector3** object is created, and returned.

Public Member Functions

Vector3 &	**operator/=** (float scalar) */= operator overloaded for scalar division*
Vector3	**operator/** (float scalar) const */ operator overloaded for scalar division*
Vector3	**operator-** () const *Negation operation overloaded.*
Vector3 &	**operator-=** (const **Vector3** &rhs) *-= operator overloaded*
Vector3 &	**operator+=** (const **Vector3** &rhs) *+= operator overloaded*
Vector3	**operator-** (const **Vector3** &rhs) const *subtraction operator overloaded*
Vector3	**operator+** (const **Vector3** &rhs) const *+ operator overloaded*
Vector3 &	**normalize** (float tolerance=1e-06) *Unitize the current vector.*
float	**lengthSqr** () const *Computes square of magnitude of the vector.*
float	**length** () const *Computes magnitude of the vector.*
Vector3 &	**operator *=** (const float scalar) **= operator for scalar multiplication*
Vector3	**operator *** (const float scalar) const ** operator for scalar multiplication*
Vector3	**unitCross** (const **Vector3** &rhs) const *Cross Product with a unit vector output.*
Vector3	**cross** (const **Vector3** &rhs) const *Cross Product.*
float	**dot** (const **Vector3** &rhs) const *Dot Product.*
bool	**operator!=** (const **Vector3** &rhs) const *!= operator overloaded*
bool	**operator==** (const **Vector3** &rhs) const *== operator overloaded*
Vector3 &	**operator=** (const **Vector3** &rhs) *= operator overloaded*

	Vector3 (const **Vector3** &pCopy) *Copy Constructor.*
	Vector3 (const float ix, const float iy, const float iz) *Additional Constructor.*
	Vector3 () *Default Constructor.*
	~Vector3 () *Destructor.*
	operator float * () *float* type cast operator*
	operator const float * () const *const float* type cast operator*

Public Attributes

float	**x** *x component in 3D space*
float	**y** *y component in 3D space*
float	**z** *z component in 3D space*

Member Function Documentation

Vector3 & XMath::Vector3::normalize (float *tolerance* = 1e-06) [ir

Makes the current Vector a unit vector.

Parameters:

tolerance	If the actual magnitude is smaller than this value, unitizing cannot be done, and a value of 0 magnitude is returned.

Returns:

Returns the old magnitude (or 0 if too small before) of the vector

Vector3 XMath::Vector3::unitCross (const Vector3 & *rhs*) const

A special version of cross multiplicatio that returns a unit vector as a result. Might be handy for lighting calculations especially.

Returns:

A unit vector perpendicular to the plane defined by this vector and the vector passed as parameter.

The documentation for this class was generated from the following file:

- Vector3.h

Xface Core Library Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

- **XEngine::MorphChannel::destroy_functor**
- **XEngine::MorphChannel::smooth_functor**
- **XEngine::ColorRGBA**
- **XEngine::Drawable**
- **XEngine::Entity**
- **XEngine::IModelLoader**
  - **XEngine::OBJLoader**
  - **XEngine::VRML1Loader**
  - **XEngine::VRML97Loader**
- **XEngine::IRenderer**
  - **XEngine::RendererGL**
- **XEngine::ITextureFile**
  - **XEngine::BmpFile**
  - **XEngine::TgaFile**
- **XEngine::ITextureLoader**
  - **XEngine::TextureLoaderGL**
- **XEngine::ITimedController**
  - **XEngine::MorphController**
- **XEngine::ITimer**
- **XEngine::KeyframeInterpolator**
- **XEngine::MeshInfo**
- **XEngine::MeshManager**
- **XEngine::ModelFileFactory**
- **XEngine::MorphChannel::dump_functor**
- **XEngine::NamedObj**
  - **XEngine::Geometry**
    - **XEngine::IndexedFaceSet**
      - **XEngine::DeformableGeometry**
  - **XEngine::ITexture**
    - **XEngine::Texture2D**
- **XEngine::RenderList**
- **XEngine::RenderManager**
- **XEngine::TextureManager**
- **XEngine::Transform**

- XEngine::Vertex2D
- XFace::FaceBase
- XFace::FAPU
- XFace::FDP
- XFace::FDPFile
- XFace::FDPItem
- XFace::FDPLoader
- XFace::IFapStream
  - XFace::FAPFile
- XFace::IInfluenceCalculator
  - XFace::RaisedCosInfluence
    - XFace::RaisedCosInfluenceSph
    - XFace::RaisedCosInfluenceWaveX
    - XFace::RaisedCosInfluenceWaveY
    - XFace::RaisedCosInfluenceWaveZ
- XFace::InfluenceCalculatorMaker
  - XFace::RaisedCosInfluenceSphMaker
  - XFace::RaisedCosInfluenceWaveXMaker
  - XFace::RaisedCosInfluenceWaveYMaker
  - XFace::RaisedCosInfluenceWaveZMaker
- XFace::XercesString
- XFaceApp::ISound
- XFaceApp::ModelCamera
- XFaceApp::Notification
- XFaceApp::Task
- XFaceApp::TaskDictionary
- XFaceApp::TaskHandlerBase
  - XFaceApp::ApplicationBase
- XFaceApp::XMLUtils
- XMath::LineSegment3
- XMath::Matrix4
- XMath::Quaternion
- XMath::Ray3
- XMath::Rectangle2
- XMath::Triangle3
- XMath::Vector3

Here is a list of all documented class members with links to the class documentation for each member:

- a -

- addAOIIndex() : **XFace::FDPItem**
- addDrawable() : **XEngine::RenderList**
- addDrawables() : **XEngine::RenderList**

Xface Core Library Namespace List

Here is a list of all documented namespaces with brief descriptions:

XEngine	
XFace	
XFaceApp	*Application related classes are under this namespace*
XMath	

Generated on Mon Aug 28 15:39:25 2006 for Xface Core Library by  *1.4.6-NO*

XEngine Namespace Reference

Detailed Description

3D engine go under this namespace.

Classes

class	**BmpFile** *BMP file loader for textures. More...*
class	**DeformableGeometry** *Deformation enabled geometry class. More...*
class	**Drawable** *Stores the information for drawable objects. More...*
class	**Entity** *Represents an entity visible in the scene, includes drawables, ai, etc.. More...*
class	**Geometry** *Defines interface for the mesh geometry. More...*
class	**IModelLoader** *Base class for the scene graph loaders. More...*
class	**IndexedFaceSet** *Indexed mesh geometry implemented. More...*
class	**IRenderer** *Interface for renderer. More...*
class	**ITexture** *Base class for texture mapping. More...*
class	**ITextureFile** *Interface for image file loaders for textures. More...*
class	**ITextureLoader** *Base class for texture loading. More...*
class	**ITimedController** *ABS for timed controllers. More...*
class	**ITimer** *Timer interface. More...*
class	**KeyframeInterpolator** *Keyframe interpolator. More...*
class	**MeshManager** *Meshes are stored and managed here. More...*
struct	**MeshInfo** *3D Meshe info More...*
class	**ModelFileFactory** *3D Meshes are loaded from here. More...*
class	**MorphController** *Morph targets controller. More...*

class	**NamedObj** *Named classes base. More...*
class	**OBJLoader** *Loads Wavefront OBJ files for the scene graph. More...*
class	**RendererGL** *OpenGL renderer. More...*
class	**RenderList** *Render list storing Drawables and calling proper **IRenderer** methods. More...*
class	**RenderManager** *Manages creation of renderer, storage of drawables to be rendered and rendering them. More...*
class	**Texture2D** *2D texture mapping More...*
class	**TextureLoaderGL** *Texture loader class for OpenGL. More...*
class	**TextureManager** *Textures are stored and managed here. More...*
class	**TgaFile** *TGA file loader for textures. More...*
class	**Transform** ***Transform** node for scenegraph. More...*
class	**Vertex2D**
class	**ColorRGBA**
class	**VRML1Loader** *VRML1 Loader. More...*
class	**VRML97Loader** *Loads VRML97 files. More...*

XFace Namespace Reference

Detailed Description

This is the main namespace **XFace** that the whole face library resides.

Classes

class	**FaceBase** *Main **XFace** class. More...*
class	**FAPFile** *MPEG-4 FAPs loader from disk. More...*
class	**FAPU** *MPEG4 **FAPU** info is in here. More...*
class	**FDP** *MPEG4 **FDP** data as a whole. More...*
class	**FDPFile** *MPEG-4 **FDP** loader/writer from/to disk. More...*
class	**FDPItem** *MPEG4 **FDP** data. More...*
class	**FDPLoader** ***FDP** Loader. More...*
class	**IFapStream** *Interface for MPEG-4 FAPs streaming. More...*
class	**IInfluenceCalculator** *Interface for defining the weights in the zone of influence for deformation. More...*
class	**InfluenceCalculatorMaker** *InfluenceCalculator pluggable factory maker. More...*
class	**RaisedCosInfluence** *Raised cosine deformation base. More...*
class	**RaisedCosInfluenceSph** *Raised cosine deformation as a sphere. More...*
class	**RaisedCosInfluenceSphMaker** *Raised cosine influence maker. More...*
class	**RaisedCosInfluenceWaveX** *Raised cosine deformation as a wave in X direction. More...*
class	**RaisedCosInfluenceWaveXMaker** *Raised cosine influence maker. More...*
class	**RaisedCosInfluenceWaveY** *Raised cosine deformation as a wave in Y direction. More...*
class	**RaisedCosInfluenceWaveYMaker** *Raised cosine Influence maker. More...*
class	**RaisedCosInfluenceWaveZ** *Raised cosine deformation as a wave in Z direction. More...*

class	**RaisedCosInfluenceWaveZMaker** *Raised cosine Influence maker. More...*
class	**XercesString** *Xerces string operations. More...*

XFaceApp Namespace Reference

Detailed Description

Author:
Koray Balci

Version:
1.0

Date:
May 2003

Classes

class	**ApplicationBase** *Base for application dev classes. More...*
class	**ISound** *Sound player interface. More...*
class	**ModelCamera** *Camera class. More...*
class	**Notification** ***Notification** message to server for remote communication. More...*
class	**TaskDictionary** *Dictionary of available Task's. More...*
class	**Task** ***Task** abstraction. More...*
class	**TaskHandlerBase** ***Task** handler base. More...*
class	**XMLUtils** *XML related functions. More...*

XMath Namespace Reference

Detailed Description

This namespace is separate from **XFace** namespace and provides general math functionality.

Classes

class	**Matrix4** *4x4 Matrix class for float More...*
class	**Quaternion** ***Quaternion** class. More...*
class	**Ray3** *3D ray representation More...*
class	**Rectangle2** *2D rectangle representation More...*
class	**Vector3** *3D Vector class More...*
class	**LineSegment3** *Representation of a 3D line segment. More...*
class	**Triangle3** *3D triangle representation More...*

Xface Core Library Modules

Here is a list of all modules:

- **XEngine**
- **XFace**
- **XMath**

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XEngine

Detailed Description

XEngine is the 3D engine used by **XFace** library. It is a simple implementation, specialized for **XFace**, implemented as generic as possible.

Classes

class	**XEngine::BmpFile** *BMP file loader for textures. More...*
class	**XEngine::DeformableGeometry** *Deformation enabled geometry class. More...*
class	**XEngine::Drawable** *Stores the information for drawable objects. More...*
class	**XEngine::Geometry** *Defines interface for the mesh geometry. More...*
class	**XEngine::IModelLoader** *Base class for the scene graph loaders. More...*
class	**XEngine::IndexedFaceSet** *Indexed mesh geometry implemented. More...*
class	**XEngine::IRenderer** *Interface for renderer. More...*
class	**XEngine::ITexture** *Base class for texture mapping. More...*
class	**XEngine::ITextureFile** *Interface for image file loaders for textures. More...*
class	**XEngine::ITextureLoader** *Base class for texture loading. More...*
class	**XEngine::ITimer** *Timer interface. More...*
class	**XEngine::MeshManager** *Meshes are stored and managed here. More...*
struct	**XEngine::MeshInfo** *3D Meshe info More...*
class	**XEngine::ModelFileFactory** *3D Meshes are loaded from here. More...*
class	**XEngine::NamedObj** *Named classes base. More...*
class	**XEngine::OBJLoader** *Loads Wavefront OBJ files for the scene graph. More...*
class	**XEngine::RendererGL** *OpenGL renderer. More...*
class	**XEngine::RenderList** *Render list storing Drawables and calling proper **IRenderer** methods. More...*

class	**XEngine::Texture2D** *2D texture mapping More...*
class	**XEngine::TextureLoaderGL** *Texture loader class for OpenGL. More...*
class	**XEngine::TextureManager** *Textures are stored and managed here. More...*
class	**XEngine::TgaFile** *TGA file loader for textures. More...*
class	**XEngine::Transform** ***Transform** node for scenegraph. More...*
class	**XEngine::Vertex2D**
class	**XEngine::ColorRGBA**
class	**XEngine::VRML1Loader** *VRML1 Loader. More...*
class	**XEngine::VRML97Loader** *Loads VRML97 files. More...*

XFace

Detailed Description

All face related **XFace** classes under **XFace** namespace are in this module.

Classes

class	**XFace::FaceBase** *Main **XFace** class. More...*
class	**XFace::FAPFile** *MPEG-4 FAPs loader from disk. More...*
class	**XFace::FAPU** *MPEG4 **FAPU** info is in here. More...*
class	**XFace::FDP** *MPEG4 **FDP** data as a whole. More...*
class	**XFace::FDPFile** *MPEG-4 **FDP** loader/writer from/to disk. More...*
class	**XFace::FDPItem** *MPEG4 **FDP** data. More...*
class	**XFace::FDPLoader** ***FDP** Loader. More...*
class	**XFace::IFapStream** *Interface for MPEG-4 FAPs streaming. More...*
class	**XFace::IInfluenceCalculator** *Interface for defining the weights in the zone of influence for deformation. More...*
class	**XFace::InfluenceCalculatorMaker** *InfluenceCalculator pluggable factory maker. More...*
class	**XFace::RaisedCosInfluence** *Raised cosine deformation base. More...*
class	**XFace::RaisedCosInfluenceSph** *Raised cosine deformation as a sphere. More...*
class	**XFace::RaisedCosInfluenceSphMaker** *Raised cosine influence maker. More...*
class	**XFace::RaisedCosInfluenceWaveX** *Raised cosine deformation as a wave in X direction. More...*
class	**XFace::RaisedCosInfluenceWaveXMaker** *Raised cosine influence maker. More...*
class	**XFace::RaisedCosInfluenceWaveY** *Raised cosine deformation as a wave in Y direction. More...*
class	**XFace::RaisedCosInfluenceWaveYMaker** *Raised cosine Influence maker. More...*
class	**XFace::RaisedCosInfluenceWaveZ** *Raised cosine deformation as a wave in Z direction. More...*

class	**XFace::RaisedCosInfluenceWaveZMaker** *Raised cosine Influence maker. More...*
class	**XFace::XercesString** *Xerces string operations. More...*

XMath

Detailed Description

Math related classes are in this module.

Classes

class	**XMath::Matrix4** *4x4 Matrix class for float More...*
class	**XMath::Ray3** *3D ray representation More...*
class	**XMath::Rectangle2** *2D rectangle representation More...*
class	**XMath::Vector3** *3D Vector class More...*
class	**XMath::LineSegment3** *Representation of a 3D line segment. More...*
class	**XMath::Triangle3** *3D triangle representation More...*

Here is a list of all documented file members with links to the documentation:

- INFINITY : **Vector3.h**

Xface Core Library Related Pages

Here is a list of all related documentation pages:

- **Skinning Adventures / Architecture Review**
- **Todo List**

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

Skinning Adventures / Architecture Review

Developer Diary

Date:
Wednesday, September 15, 2004

I finally finished working on skinning. First of all, I didn't really implement the conventional vertex skinning, because I realized that I don't have to!

I was planning to use ARB_vertex_blend OpenGL extension and implement a mid-level path along with hw shader and sw deformer. Unfortunately, I realized that, this path requires some change in the mesh data structure. I needed to have submeshes connected to all deformers, something that I don't want to implement really. Actually, my current structure is much better than that already, and also both hw and sw paths do not need that kind of structuring. Alternatively, you can use ARB_matrix_palette extension along with vertex_blend to overcome this, but it is not supported by any of the card vendors anymore! People say, this is because skinning by extensions is replaced by hw shader versions. So, as a result, I also decided to forget about this path totally.

As soon as I gave up that path, I realized that I do NOT need to change my architecture that much, I do NOT need to implement skinning in the way it is explained. I don't need to store transformation matrices and bone structures for every deformation point as I planned. Here is my architecture.

XFace::FaceBase class controls the whole system as before. We have a new class for geometry to handle the deformations, **XEngine::DeformableGeometry**. Each **XEngine::Drawable** object now has a **XEngine::DeformableGeometry** bound to it. As before, **XEngine::Drawable**'s represent the building blocks of the head, i.e. the face, hair, tongue, etc.. FaceBase, has a list of **XEngine::Drawable** objects, during the initialization phase, the meshes are loaded from disk into **XEngine::DeformableGeometry** objects and attached to **XEngine::Drawable** objects. In reality, only the names of **XEngine::DeformableGeometry** objects are stored in

XEngine::Drawable, and they are retrieved through **XEngine::MeshManager** singleton object. During the initialization phase, also FDP file is loaded and XFace::FDPItems are filled. **XFace::FDPItem** objects represent MPEG-4 feature points defined by the standard, and store the characteristics of the feature point. Namely, we store, for each feature point;

the mesh it is affecting, control point that represents the feature point, the region (vertices) influenced by the control point, a set of influence calculators

Influence calculators store weights for each vertex in the zone of influence, where weight represents how the coresponding vertex will be affected by the deformation of that feature point. Interface for the influence calculators are defined by XFace::IInfluenceCalculators class. Currently, we have a set of raised cosine based influence calculator concrete classes. XfaceEd automatically detects influence calculators, thanks to pluggable factory pattern implemented inside. See **XFace::InfluenceCalculatorMaker** for more information on that. Each **XFace::IInfluenceCalculator** derived object also stores a coefficient to control the deformation limits, and the FAP id that the weights are used for (or in other words, the FAP that the influence calculator responds to).

So during the initialization, when the FDP file is loaded, first static meshes (OBJ/VRML/X3D or any other supported format) are loaded to **XEngine::DeformableGeometry** instances, and feature point info are loaded to XFace::FDPItems, and XFace::IInfluenceCalculators are created and the weights for the zone of influence are calculated. Then those weights are routed to **XEngine::DeformableGeometry** objects.

XEngine::DeformableGeometry objects store two sets of vertices, one for the still (static, no emotion/movement) and one for the deformed. During runtime, when playback of FAPs starts, for each new frame, 68 FAP values are sent to XEngine::DeformableGeometry::updateAnimationParams method, but the deformation is delayed until the rendering. During the rendering process, if a geometry needs an update, its update method is called for the moment, though I plan to move this piece of code out of **XEngine::DeformableGeometry**. This lazy evaluation (in the rendering

pipeline) enables us to call a shader instead of updating on software. Though I haven't implemented the shader pipeline, all the structuring is done.

Todo List

Member **XEngine::DeformableGeometry::update** (const std::vector< float > &aps)

computeVertexNormals() ??

Member **XEngine::Geometry::setNormals** (const Vector3 *pNorm, unsigned int size)

check back here

Member **XEngine::Geometry::setVertices** (const Vector3 *pVert, unsigned int size)

check back here

Class **XEngine::ModelFileFactory**

I guess this is only necessary for XFaceEd, so we might consider moving this class there.

Class **XEngine::VRML97Loader**

Not finished yet, need to implement full scene graph struct for proper use.

Member **XFaceApp::ApplicationBase::query** (const QUERY &q)

The mechanism can be improved and reverted to Task/Notification type easily. Or at least we can add a new Task type for QUERY and implement handlers.

Member **XFaceApp::ApplicationBase::synchronize** (bool)

Check this function, not tested extensively. Especially in debug mode, we seem to have some problem, should skip the frames, but not as it seems, lagging animation.

Class **XFaceApp::ModelCamera**

This class should be in another name space, in some other place, **XEngine** perhaps?

Class **XFaceApp::TaskDictionary**

Should implement a better way to upload binary files first to implement this (UPLOAD_WAV) Task, note that it is also necessary

to be able to upload binary (compressed) FAP files.

- **UPLOAD_ANIM** UpLoads ANIM file created by SMIL-Agent (temporary addition)

XEngine::MorphChannel::destroy_functor Member List

This is the complete list of members for **XEngine::MorphChannel::destroy_functor**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XEngine::MorphChannel::smooth_functor Member List

This is the complete list of members for **XEngine::MorphChannel::smooth_functor**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XEngine::BmpFile Member List

This is the complete list of members for **XEngine::BmpFile**, including all inherited members.

getData()	**XEngine::ITextureFile**	[inline]
getHeight() const	**XEngine::ITextureFile**	[inline]
getNBits() const	**XEngine::ITextureFile**	[inline]
getWidth() const	**XEngine::ITextureFile**	[inline]
load(const std::string &filename)	**XEngine::BmpFile**	[virtual]
m_Height	**XEngine::ITextureFile**	[protected]
m_nBits	**XEngine::ITextureFile**	[protected]
m_pData	**XEngine::ITextureFile**	[protected]
m_Width	**XEngine::ITextureFile**	[protected]

XEngine::ColorRGBA Member List

This is the complete list of members for **XEngine::ColorRGBA**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XEngine::DeformableGeometry Member List

This is the complete list of members for **XEngine::DeformableGeometry**, including all inherited members.

computeFaceNormal(const Vector3 &p1, const Vector3 &p2, const Vector3 &p3)	**XEngine::Geometry**	[static]
computeVertexNormals()	**XEngine::IndexedFaceSet**	
XEngine::IndexedFaceSet::copyFrom(const IndexedFaceSet &rhs)	**XEngine::IndexedFaceSet**	
XEngine::Geometry::copyFrom(const Geometry &rhs)	**XEngine::Geometry**	[virtual]
getIndexCount() const	**XEngine::IndexedFaceSet**	[inline]
getIndices() const	**XEngine::IndexedFaceSet**	[inline]
getName() const	**XEngine::NamedObj**	[inline]
getNormals() const	**XEngine::Geometry**	[inline]
getTexCoords() const	**XEngine::Geometry**	[inline]
getVertexCount() const	**XEngine::Geometry**	[inline]
getVertices() const	**XEngine::Geometry**	[inline]
NamedObj(const std::string &name)	**XEngine::NamedObj**	[inline, protected]
setIndices(const std::vector< std::vector< unsigned short > > &indices)	**XEngine::IndexedFaceSet**	
setName(const std::string &name)	**XEngine::NamedObj**	[inline]
setNormals(const std::vector< Vector3 > &normals)	**XEngine::Geometry**	[virtual]
setNormals(const Vector3 *pNorm, unsigned int size)	**XEngine::Geometry**	[virtual]
setTexCoords(const Vertex2D *pTex, unsigned int size)	**XEngine::Geometry**	[virtual]
setTexCoords(const std::vector< Vertex2D > &texCoords)	**XEngine::Geometry**	[virtual]
setVertices(const std::vector< Vector3 > &vertices)	**XEngine::DeformableGeometry**	[virtual]
XEngine::IndexedFaceSet::setVertices(const Vector3 *pVert, unsigned int size)	**XEngine::Geometry**	[virtual]
update(const std::vector< float > &aps)	**XEngine::DeformableGeometry**	
~NamedObj(void)	**XEngine::NamedObj**	[inline, protected, virtual]

XEngine::Drawable Member List

This is the complete list of members for **XEngine::Drawable**, including all inherited members.

clone(bool duplicateData=true) const	**XEngine::Drawable**	[inline]
Drawable()	**XEngine::Drawable**	[inline]
enableTexture(bool mode)	**XEngine::Drawable**	[inline]
getMeshName() const	**XEngine::Drawable**	[inline]
getTexName(unsigned short layer=0) const	**XEngine::Drawable**	[inline]
isTextureOn() const	**XEngine::Drawable**	[inline]
setMeshName(const std::string &MeshName)	**XEngine::Drawable**	[inline]
setTexName(const std::string id, unsigned short layer=0)	**XEngine::Drawable**	[inline]
~Drawable()	**XEngine::Drawable**	[inline]

XEngine::Entity Member List

This is the complete list of members for **XEngine::Entity**, including all inherited members.

Generated on Mon Aug 28 15:39:25 2006 for Xface Core Library by  *1.4.6-NO*

XEngine::Geometry Member List

This is the complete list of members for **XEngine::Geometry**, including all inherited members.

computeFaceNormal(const Vector3 &p1, const Vector3 &p2, const Vector3 &p3)	**XEngine::Geometry**	[static]
copyFrom(const Geometry &rhs)	**XEngine::Geometry**	[virtual]
getName() const	**XEngine::NamedObj**	[inline]
getNormals() const	**XEngine::Geometry**	[inline]
getTexCoords() const	**XEngine::Geometry**	[inline]
getVertexCount() const	**XEngine::Geometry**	[inline]
getVertices() const	**XEngine::Geometry**	[inline]
NamedObj(const std::string &name)	**XEngine::NamedObj**	[inline, protected]
setName(const std::string &name)	**XEngine::NamedObj**	[inline]
setNormals(const std::vector< Vector3 > &normals)	**XEngine::Geometry**	[virtual]
setNormals(const Vector3 *pNorm, unsigned int size)	**XEngine::Geometry**	[virtual]
setTexCoords(const Vertex2D *pTex, unsigned int size)	**XEngine::Geometry**	[virtual]
setTexCoords(const std::vector< Vertex2D > &texCoords)	**XEngine::Geometry**	[virtual]
setVertices(const Vector3 *pVert, unsigned int size)	**XEngine::Geometry**	[virtual]
setVertices(const std::vector< Vector3 > &vertices)	**XEngine::Geometry**	[virtual]
~NamedObj(void)	**XEngine::NamedObj**	[inline, protected, virtual]

XEngine::IModelLoader Member List

This is the complete list of members for **XEngine::IModelLoader**, including all inherited members.

loadModel(const std::string &filename, const std::string &dir="./")=0	**XEngine::IModelLoader**	[pure virtual]

XEngine::IndexedFaceSet Member List

This is the complete list of members for **XEngine::IndexedFaceSet**, including all inherited members.

computeFaceNormal(const Vector3 &p1, const Vector3 &p2, const Vector3 &p3)	**XEngine::Geometry**	[static]
computeVertexNormals()	**XEngine::IndexedFaceSet**	
copyFrom(const IndexedFaceSet &rhs)	**XEngine::IndexedFaceSet**	
XEngine::Geometry::copyFrom(const Geometry &rhs)	**XEngine::Geometry**	[virtual]
getIndexCount() const	**XEngine::IndexedFaceSet**	[inline]
getIndices() const	**XEngine::IndexedFaceSet**	[inline]
getName() const	**XEngine::NamedObj**	[inline]
getNormals() const	**XEngine::Geometry**	[inline]
getTexCoords() const	**XEngine::Geometry**	[inline]
getVertexCount() const	**XEngine::Geometry**	[inline]
getVertices() const	**XEngine::Geometry**	[inline]
NamedObj(const std::string &name)	**XEngine::NamedObj**	[inline, protected]
setIndices(const std::vector< std::vector< unsigned short > > &indices)	**XEngine::IndexedFaceSet**	
setName(const std::string &name)	**XEngine::NamedObj**	[inline]
setNormals(const std::vector< Vector3 > &normals)	**XEngine::Geometry**	[virtual]
setNormals(const Vector3 *pNorm, unsigned int size)	**XEngine::Geometry**	[virtual]
setTexCoords(const Vertex2D *pTex, unsigned int size)	**XEngine::Geometry**	[virtual]
setTexCoords(const std::vector< Vertex2D > &texCoords)	**XEngine::Geometry**	[virtual]
setVertices(const Vector3 *pVert, unsigned int size)	**XEngine::Geometry**	[virtual]
setVertices(const std::vector< Vector3 > &vertices)	**XEngine::Geometry**	[virtual]
~NamedObj(void)	**XEngine::NamedObj**	[inline, protected, virtual]

XEngine::IRenderer Member List

This is the complete list of members for **XEngine::IRenderer**, including all inherited members.

render(Drawable *const pDrawable) const =0	**XEngine::IRenderer**	`[pure virtual]`

XEngine::ITexture Member List

This is the complete list of members for **XEngine::ITexture**, including all inherited members.

getName() const	**XEngine::NamedObj**	[inline]
getTextureID() const	**XEngine::ITexture**	[inline]
getTextureType() const =0	**XEngine::ITexture**	[pure virtual]
ITexture(const std::string &name)	**XEngine::ITexture**	[inline]
NamedObj(const std::string &name)	**XEngine::NamedObj**	[inline, protected]
setName(const std::string &name)	**XEngine::NamedObj**	[inline]
TEXTURE2D enum value	**XEngine::ITexture**	
TEXTURETYPE enum name	**XEngine::ITexture**	
~ITexture()	**XEngine::ITexture**	[inline, virtual]
~NamedObj(void)	**XEngine::NamedObj**	[inline, protected, virtual]

XEngine::ITextureFile Member List

This is the complete list of members for **XEngine::ITextureFile**, including all inherited members.

getData()	**XEngine::ITextureFile**	[inline]
getHeight() const	**XEngine::ITextureFile**	[inline]
getNBits() const	**XEngine::ITextureFile**	[inline]
getWidth() const	**XEngine::ITextureFile**	[inline]
load(const std::string &filename)=0	**XEngine::ITextureFile**	[pure virtual]
m_Height	**XEngine::ITextureFile**	[protected]
m_nBits	**XEngine::ITextureFile**	[protected]
m_pData	**XEngine::ITextureFile**	[protected]
m_Width	**XEngine::ITextureFile**	[protected]

XEngine::ITextureLoader Member List

This is the complete list of members for **XEngine::ITextureLoader**, including all inherited members.

ITextureLoader()	**XEngine::ITextureLoader**	`[inline]`
load(const std::string &filename, ITexture *pTexture)=0	**XEngine::ITextureLoader**	`[pure virtual]`
unLoad(const ITexture *pTexture)=0	**XEngine::ITextureLoader**	`[pure virtual]`
~ITextureLoader()	**XEngine::ITextureLoader**	`[inline, virtual]`

XEngine::ITimedController Member List

This is the complete list of members for **XEngine::ITimedController**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XEngine::ITimer Member List

This is the complete list of members for **XEngine::ITimer**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XEngine::KeyframeInterpolator Member List

This is the complete list of members for **XEngine::KeyframeInterpolator**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XEngine::MeshInfo Member List

This is the complete list of members for **XEngine::MeshInfo**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XEngine::MeshManager Member List

This is the complete list of members for **XEngine::MeshManager**, including all inherited members.

destroyAll()	**XEngine::MeshManager**	
getInstance()	**XEngine::MeshManager**	[static]
getMesh(const std::string &name) const	**XEngine::MeshManager**	
registerMesh(DeformableGeometry *pMesh)	**XEngine::MeshManager**	
releaseInstance()	**XEngine::MeshManager**	[static]

XEngine::ModelFileFactory Member List

This is the complete list of members for **XEngine::ModelFileFactory**, including all inherited members.

getMeshInfo(const std::string &meshName)	**XEngine::ModelFileFactory**	[static]
isFileLoaded(const std::string &filename)	**XEngine::ModelFileFactory**	[static]
loadModelFile(const std::string &filename, const std::string &path="./")	**XEngine::ModelFileFactory**	[static]
unloadAllFiles()	**XEngine::ModelFileFactory**	[static]
unloadModelFile(const std::string &filename)	**XEngine::ModelFileFactory**	[static]

XEngine::MorphChannel::dump_functor Member List

This is the complete list of members for **XEngine::MorphChannel::dump_functor**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XEngine::MorphController Member List

This is the complete list of members for **XEngine::MorphController**, including all inherited members.

getInstance()	**XEngine::MorphController**	[static]

XEngine::NamedObj Member List

This is the complete list of members for **XEngine::NamedObj**, including all inherited members.

getName() const	**XEngine::NamedObj**	`[inline]`
NamedObj(const std::string &name)	**XEngine::NamedObj**	`[inline, protected]`
setName(const std::string &name)	**XEngine::NamedObj**	`[inline]`
~NamedObj(void)	**XEngine::NamedObj**	`[inline, protected, virtual]`

XEngine::OBJLoader Member List

This is the complete list of members for **XEngine::OBJLoader**, including all inherited members.

loadModel(const std::string &filename, const std::string &dir="./")	**XEngine::OBJLoader**	`[virtual]`

XEngine::RendererGL Member List

This is the complete list of members for **XEngine::RendererGL**, including all inherited members.

render(Drawable *const pDrawable) const	**XEngine::RendererGL**	[virtual]

XEngine::RenderList Member List

This is the complete list of members for **XEngine::RenderList**, including all inherited members.

addDrawable(boost::shared_ptr< Drawable > item)	**XEngine::RenderList**	[inline]
addDrawables(const std::list< boost::shared_ptr< Drawable > > &dr)	**XEngine::RenderList**	[inline]
clearList()	**XEngine::RenderList**	[inline]
removeDrawable(const std::string &name)	**XEngine::RenderList**	
renderList(boost::shared_ptr< IRenderer > pRenderer) const	**XEngine::RenderList**	

XEngine::RenderManager Member List

This is the complete list of members for **XEngine::RenderManager**, including all inherited members.

update()	**XEngine::RenderManager**

XEngine::Texture2D Member List

This is the complete list of members for **XEngine::Texture2D**, including all inherited members.

getName() const	**XEngine::NamedObj**	[inline]
getTextureID() const	**XEngine::ITexture**	[inline]
getTextureType() const	**XEngine::Texture2D**	[inline, virtual]
ITexture(const std::string &name)	**XEngine::ITexture**	[inline]
NamedObj(const std::string &name)	**XEngine::NamedObj**	[inline, protected]
setName(const std::string &name)	**XEngine::NamedObj**	[inline]
Texture2D(const std::string &name)	**XEngine::Texture2D**	[inline]
TEXTURE2D enum value	**XEngine::ITexture**	
TEXTURETYPE enum name	**XEngine::ITexture**	
~ITexture()	**XEngine::ITexture**	[inline, virtual]
~NamedObj(void)	**XEngine::NamedObj**	[inline, protected, virtual]
~Texture2D()	**XEngine::Texture2D**	[inline, virtual]

XEngine::TextureLoaderGL Member List

This is the complete list of members for **XEngine::TextureLoaderGL**, including all inherited members.

ITextureLoader()	**XEngine::ITextureLoader**	[inline]
load(const std::string &filename, ITexture *pTexture)	**XEngine::TextureLoaderGL**	[virtual]
TextureLoaderGL()	**XEngine::TextureLoaderGL**	
unLoad(const ITexture *pTexture)	**XEngine::TextureLoaderGL**	[virtual]
~ITextureLoader()	**XEngine::ITextureLoader**	[inline, virtual]
~TextureLoaderGL()	**XEngine::TextureLoaderGL**	[virtual]

XEngine::TextureManager Member List

This is the complete list of members for **XEngine::TextureManager**, including all inherited members.

destroyAll()	**XEngine::TextureManager**	
getInstance()	**XEngine::TextureManager**	[static]
getTexture(const std::string &name) const	**XEngine::TextureManager**	
load(const std::string &filename, const std::string &texname)	**XEngine::TextureManager**	
registerTexture(const ITexture *pTexture)	**XEngine::TextureManager**	
releaseInstance()	**XEngine::TextureManager**	[static]
unLoad(const ITexture *pTexture)	**XEngine::TextureManager**	
unLoad(const std::string &name)	**XEngine::TextureManager**	
unregisterTexture(const ITexture *pTexture)	**XEngine::TextureManager**	

XEngine::TgaFile Member List

This is the complete list of members for **XEngine::TgaFile**, including all inherited members.

getData()	**XEngine::ITextureFile**	[inline]
getHeight() const	**XEngine::ITextureFile**	[inline]
getNBits() const	**XEngine::ITextureFile**	[inline]
getWidth() const	**XEngine::ITextureFile**	[inline]
load(const std::string &filename)	**XEngine::TgaFile**	[virtual]
m_Height	**XEngine::ITextureFile**	[protected]
m_nBits	**XEngine::ITextureFile**	[protected]
m_pData	**XEngine::ITextureFile**	[protected]
m_Width	**XEngine::ITextureFile**	[protected]

XEngine::Transform Member List

This is the complete list of members for **XEngine::Transform**, including all inherited members.

getRotation() const	**XEngine::Transform**	[inline]
getTranslation() const	**XEngine::Transform**	[inline]
setRotation(const Quaternion &qRot)	**XEngine::Transform**	[inline]
setScale(const Vector3 &scale)	**XEngine::Transform**	[inline]
setTranslation(const Vector3 &trans)	**XEngine::Transform**	[inline]
setTranslation(float x, float y, float z)	**XEngine::Transform**	[inline]

XEngine::Vertex2D Member List

This is the complete list of members for **XEngine::Vertex2D**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XEngine::VRML1Loader Member List

This is the complete list of members for **XEngine::VRML1Loader**, including all inherited members.

loadModel(const std::string &filename, const std::string &dir="./")	**XEngine::VRML1Loader**	`[virtual]`

XEngine::VRML97Loader Member List

This is the complete list of members for **XEngine::VRML97Loader**, including all inherited members.

loadModel(const std::string &filename, const std::string &dir="./")	**XEngine::VRML97Loader**	`[virtual]`

XFace::FaceBase Member List

This is the complete list of members for **XFace::FaceBase**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XFace::FAPFile Member List

This is the complete list of members for **XFace::FAPFile**, including all inherited members.

getCurrentFAP() const	**XFace::IFapStream**	[inline]
getFAPCount() const	**XFace::FAPFile**	[inline, virtual]
getFPS() const	**XFace::IFapStream**	[inline]
getVersion() const	**XFace::IFapStream**	[inline]
isEnd() const	**XFace::FAPFile**	[inline, virtual]
isOpen() const	**XFace::FAPFile**	[inline, virtual]
m_currentFAP	**XFace::IFapStream**	[protected]
m_FPS	**XFace::IFapStream**	[protected]
m_version	**XFace::IFapStream**	[protected]
next()	**XFace::FAPFile**	[virtual]
open(std::istream &input, const FAPU &fapu)	**XFace::FAPFile**	[virtual]
rewind()	**XFace::FAPFile**	[virtual]
save(IFapStream &faps, const FAPU &fapu, const std::string &filename)	**XFace::FAPFile**	[static]

XFace::FAPU Member List

This is the complete list of members for **XFace::FAPU**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XFace::FDP Member List

This is the complete list of members for **XFace::FDP**, including all inherited members.

findItem(const std::string &name, const std::string &aff) const	**XFace::FDP**	
getENS() const	**XFace::FDP**	[inline]
getES() const	**XFace::FDP**	[inline]
getFAPU() const	**XFace::FDP**	[inline]
getGlobalAxisAngle() const	**XFace::FDP**	[inline]
getGlobalRotationAngle() const	**XFace::FDP**	[inline]
getGlobalRotationAxis() const	**XFace::FDP**	[inline]
getGlobalTranslation() const	**XFace::FDP**	[inline]
getIRISD() const	**XFace::FDP**	[inline]
getItemCount() const	**XFace::FDP**	[inline]
getItems() const	**XFace::FDP**	[inline]
getMNS() const	**XFace::FDP**	[inline]
getMW() const	**XFace::FDP**	[inline]
insertItem(FDPItem *item)	**XFace::FDP**	[inline]
setENS0(float ens)	**XFace::FDP**	[inline]
setES0(float es)	**XFace::FDP**	[inline]
setFAPU(const FAPU &fapus)	**XFace::FDP**	[inline]
setGlobalRotation(const AxisAngle &axisAng)	**XFace::FDP**	[inline]
setGlobalRotation(float x, float y, float z, float angle)	**XFace::FDP**	[inline]
setGlobalTranslation(const Vector3 &tr)	**XFace::FDP**	[inline]
setGlobalTranslation(float x, float y, float z)	**XFace::FDP**	[inline]
setIRISD0(float irisd)	**XFace::FDP**	[inline]
setMNS0(float mns)	**XFace::FDP**	[inline]
setMW0(float mw)	**XFace::FDP**	[inline]

XFace::FDPFile Member List

This is the complete list of members for **XFace::FDPFile**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XFace::FDPItem Member List

This is the complete list of members for **XFace::FDPItem**, including all inherited members.

addAOIIndex(unsigned short ind)	**XFace::FDPItem**	[inline]
getAffects() const	**XFace::FDPItem**	[inline]
getAOI() const	**XFace::FDPItem**	[inline]
getAOICount() const	**XFace::FDPItem**	[inline]
getIndex() const	**XFace::FDPItem**	[inline]
getName() const	**XFace::FDPItem**	[inline]
hasControlPoint() const	**XFace::FDPItem**	[inline]
modifyInfluenceCalculator(unsigned int order, const std::string &type, float w, unsigned short fap)	**XFace::FDPItem**	
removeAOIIndex(unsigned short ind)	**XFace::FDPItem**	[inline]
removeInfluenceCalculator(const std::string &type, float w, unsigned short id)	**XFace::FDPItem**	
resetAOI()	**XFace::FDPItem**	[inline]
setAffects(const std::string &aff)	**XFace::FDPItem**	[inline]
setControlPoint(unsigned short ind)	**XFace::FDPItem**	[inline]
setName(const std::string &name)	**XFace::FDPItem**	[inline]

XFace::FDPLoader Member List

This is the complete list of members for **XFace::FDPLoader**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XFace::IFapStream Member List

This is the complete list of members for **XFace::IFapStream**, including all inherited members.

getCurrentFAP() const	**XFace::IFapStream**	[inline]
getFAPCount() const =0	**XFace::IFapStream**	[pure virtual]
getFPS() const	**XFace::IFapStream**	[inline]
getVersion() const	**XFace::IFapStream**	[inline]
isEnd() const =0	**XFace::IFapStream**	[pure virtual]
isOpen() const =0	**XFace::IFapStream**	[pure virtual]
m_currentFAP	**XFace::IFapStream**	[protected]
m_FPS	**XFace::IFapStream**	[protected]
m_version	**XFace::IFapStream**	[protected]
next()=0	**XFace::IFapStream**	[pure virtual]
open(std::istream &, const FAPU &)=0	**XFace::IFapStream**	[pure virtual]
rewind()=0	**XFace::IFapStream**	[pure virtual]

XFace::IInfluenceCalculator Member List

This is the complete list of members for **XFace::IInfluenceCalculator**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XFace::InfluenceCalculatorMaker Member List

This is the complete list of members for **XFace::InfluenceCalculatorMaker**, including all inherited members.

getMakerList()	**XFace::InfluenceCalculatorMaker**	[static]
InfluenceCalculatorMaker(const std::string &className)	**XFace::InfluenceCalculatorMaker**	[protected]
makeInfluenceCalculator(float w, unsigned short fapID) const =0	**XFace::InfluenceCalculatorMaker**	[protected, pure virtual]
newInfluenceCalculator(const std::string &, float w, unsigned short fapID)	**XFace::InfluenceCalculatorMaker**	[static]

XFace::RaisedCosInfluence Member List

This is the complete list of members for **XFace::RaisedCosInfluence**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XFace::RaisedCosInfluenceSph Member List

This is the complete list of members for **XFace::RaisedCosInfluenceSph**, including all inherited members.

XFace::RaisedCosInfluenceSphMaker Member List

This is the complete list of members for **XFace::RaisedCosInfluenceSphMaker**, including all inherited members.

getMakerList()	**XFace::InfluenceCalculatorMaker**	`[static]`
InfluenceCalculatorMaker(const std::string &className)	**XFace::InfluenceCalculatorMaker**	`[protected]`
newInfluenceCalculator(const std::string &, float w, unsigned short fapID)	**XFace::InfluenceCalculatorMaker**	`[static]`

XFace::RaisedCosInfluenceWaveX Member List

This is the complete list of members for **XFace::RaisedCosInfluenceWaveX**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XFace::RaisedCosInfluenceWaveXMaker Member List

This is the complete list of members for **XFace::RaisedCosInfluenceWaveXMaker**, including all inherited members.

getMakerList()	**XFace::InfluenceCalculatorMaker**	`[static]`
InfluenceCalculatorMaker(const std::string &className)	**XFace::InfluenceCalculatorMaker**	`[protected]`
newInfluenceCalculator(const std::string &, float w, unsigned short fapID)	**XFace::InfluenceCalculatorMaker**	`[static]`

XFace::RaisedCosInfluenceWaveY Member List

This is the complete list of members for **XFace::RaisedCosInfluenceWaveY**, including all inherited members.

XFace::RaisedCosInfluenceWaveYMaker Member List

This is the complete list of members for **XFace::RaisedCosInfluenceWaveYMaker**, including all inherited members.

getMakerList()	**XFace::InfluenceCalculatorMaker**	[static]
InfluenceCalculatorMaker(const std::string &className)	**XFace::InfluenceCalculatorMaker**	[protected]
newInfluenceCalculator(const std::string &, float w, unsigned short fapID)	**XFace::InfluenceCalculatorMaker**	[static]

XFace::RaisedCosInfluenceWaveZ Member List

This is the complete list of members for **XFace::RaisedCosInfluenceWaveZ**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XFace::RaisedCosInfluenceWaveZMaker Member List

This is the complete list of members for **XFace::RaisedCosInfluenceWaveZMaker**, including all inherited members.

getMakerList()	**XFace::InfluenceCalculatorMaker**	`[static]`
InfluenceCalculatorMaker(const std::string &className)	**XFace::InfluenceCalculatorMaker**	`[protected]`
newInfluenceCalculator(const std::string &, float w, unsigned short fapID)	**XFace::InfluenceCalculatorMaker**	`[static]`

XFace::XercesString Member List

This is the complete list of members for **XFace::XercesString**, including all inherited members.

operator!=(const char *str) const **XFace::XercesString**

XFaceApp::ApplicationBase Member List

This is the complete list of members for **XFaceApp::ApplicationBase**, including all inherited members.

createFace() const =0	**XFaceApp::ApplicationBase**	[protected, pure virtual]
createFapStream() const =0	**XFaceApp::ApplicationBase**	[protected, pure virtual]
createRenderer() const =0	**XFaceApp::ApplicationBase**	[protected, pure virtual]
createScriptProcessors()=0	**XFaceApp::ApplicationBase**	[protected, pure virtual]
createSound() const =0	**XFaceApp::ApplicationBase**	[protected, pure virtual]
createTimer() const =0	**XFaceApp::ApplicationBase**	[protected, pure virtual]
discardPendingTasks()	**XFaceApp::TaskHandlerBase**	
fireNotification(const Notification &note) const =0	**XFaceApp::ApplicationBase**	[protected, pure virtual]
init()	**XFaceApp::ApplicationBase**	[virtual]
m_currentTask	**XFaceApp::TaskHandlerBase**	[protected]
muteAudio(bool bSnd)	**XFaceApp::ApplicationBase**	[virtual]
newTask(const Task &task)	**XFaceApp::TaskHandlerBase**	[virtual]
onAdvanceFrame()	**XFaceApp::ApplicationBase**	[protected, virtual]
XFaceApp::TaskHandlerBase::onAdvanceFrame() const	**XFaceApp::TaskHandlerBase**	[inline, protected, virtual]
onLoadANIM(const std::string &param1)	**XFaceApp::ApplicationBase**	[protected, virtual]
onLoadFAP(const std::string &param)	**XFaceApp::ApplicationBase**	[protected, virtual]
onLoadFDP(const std::string &param1, const std::string &param2)	**XFaceApp::ApplicationBase**	[protected, virtual]
onLoadPHO(const std::string &param1, const std::string &param2)	**XFaceApp::ApplicationBase**	[protected, virtual]
onLoadWAV(const std::string &param)	**XFaceApp::ApplicationBase**	[protected, virtual]
onRenderFrame() const	**XFaceApp::ApplicationBase**	[protected, virtual]
onResumePlayback()	**XFaceApp::ApplicationBase**	[protected, virtual]
		[protected,

onRewindPlayback() const	**XFaceApp::ApplicationBase**	`virtual]`
onStopPlayback()	**XFaceApp::ApplicationBase**	`[protected, virtual]`
onUpLoadANIM(const std::string &param1)	**XFaceApp::ApplicationBase**	`[protected, virtual]`
onUpLoadFAP(const std::string &param)	**XFaceApp::ApplicationBase**	`[protected, virtual]`
onUpLoadPHO(const std::string &param1, const std::string &param2, const std::string &param3)	**XFaceApp::ApplicationBase**	`[protected, virtual]`
onUpLoadScript(const std::string &param1, const std::string &param2)	**XFaceApp::ApplicationBase**	`[protected, virtual]`
pause(bool bPause=true)	**XFaceApp::TaskHandlerBase**	`[inline]`
processTask()	**XFaceApp::TaskHandlerBase**	`[virtual]`
query(const QUERY &q)	**XFaceApp::ApplicationBase**	
QUERY enum name	**XFaceApp::ApplicationBase**	
renderBegin() const	**XFaceApp::ApplicationBase**	`[inline, protected, virtual]`
renderEnd() const	**XFaceApp::ApplicationBase**	`[inline, protected, virtual]`
synchronize(bool)	**XFaceApp::ApplicationBase**	`[protected]`
yield()=0	**XFaceApp::ApplicationBase**	`[protected, pure virtual]`

XFaceApp::ISound Member List

This is the complete list of members for **XFaceApp::ISound**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XFaceApp::ModelCamera Member List

This is the complete list of members for **XFaceApp::ModelCamera**, including all inherited members.

Generated on Mon Aug 28 15:39:26 2006 for Xface Core Library by  *1.4.6-NO*

XFaceApp::Notification Member List

This is the complete list of members for **XFaceApp::Notification**, including all inherited members.

error()	**XFaceApp::Notification**	[inline, static]
getName() const	**XFaceApp::Notification**	[inline]
getOwnerID() const	**XFaceApp::Notification**	[inline]
getStatus() const	**XFaceApp::Notification**	[inline]
getTaskID() const	**XFaceApp::Notification**	[inline]
kDiscarded enum value	**XFaceApp::Notification**	
kError enum value	**XFaceApp::Notification**	
kFinished enum value	**XFaceApp::Notification**	
kInProgress enum value	**XFaceApp::Notification**	
kInQueue enum value	**XFaceApp::Notification**	
kQueued enum value	**XFaceApp::Notification**	
kStarted enum value	**XFaceApp::Notification**	
kUnknown enum value	**XFaceApp::Notification**	
Notification(const std::string &name, unsigned short owner=0, unsigned int task=0)	**XFaceApp::Notification**	
Notification(const Task &task, const TaskStatus &status)	**XFaceApp::Notification**	
setStatus(TaskStatus status)	**XFaceApp::Notification**	[inline]
TaskStatus enum name	**XFaceApp::Notification**	

XFaceApp::Task Member List

This is the complete list of members for **XFaceApp::Task**, including all inherited members.

clearParams()	**XFaceApp::Task**	[inline]
getID() const	**XFaceApp::Task**	[inline]
getName() const	**XFaceApp::Task**	[inline]
getOwnerID() const	**XFaceApp::Task**	[inline]
getParamCount() const	**XFaceApp::Task**	[inline]
getParameter(unsigned int id) const	**XFaceApp::Task**	
pushParameter(const std::string &param)	**XFaceApp::Task**	[inline]
setID(unsigned int id=0)	**XFaceApp::Task**	[inline]
setName(const std::string &name)	**XFaceApp::Task**	
setOwnerID(unsigned short id=0)	**XFaceApp::Task**	[inline]
Task(const std::string &name, unsigned short owner=0, unsigned int id=0)	**XFaceApp::Task**	[inline]

XFaceApp::TaskDictionary Member List

This is the complete list of members for **XFaceApp::TaskDictionary**, including all inherited members.

isTask(const std::string &name)	**XFaceApp::TaskDictionary**	[inline, static]
queryParamCount(const std::string &name)	**XFaceApp::TaskDictionary**	[inline, static]
registerTask(const std::string &name, int nParam)	**XFaceApp::TaskDictionary**	[inline, static]

XFaceApp::TaskHandlerBase Member List

This is the complete list of members for **XFaceApp::TaskHandlerBase**, including all inherited members.

discardPendingTasks()	**XFaceApp::TaskHandlerBase**	
m_currentTask	**XFaceApp::TaskHandlerBase**	[protected]
newTask(const Task &task)	**XFaceApp::TaskHandlerBase**	[virtual]
onAdvanceFrame() const	**XFaceApp::TaskHandlerBase**	[inline, protected, virtual]
onLoadANIM(const std::string &)	**XFaceApp::TaskHandlerBase**	[inline, protected, virtual]
onLoadFAP(const std::string &)	**XFaceApp::TaskHandlerBase**	[inline, protected, virtual]
onLoadFDP(const std::string &, const std::string &)	**XFaceApp::TaskHandlerBase**	[inline, protected, virtual]
onLoadPHO(const std::string &, const std::string &)	**XFaceApp::TaskHandlerBase**	[inline, protected, virtual]
onLoadWAV(const std::string &)	**XFaceApp::TaskHandlerBase**	[inline, protected, virtual]
onRenderFrame() const	**XFaceApp::TaskHandlerBase**	[inline, protected, virtual]
onResumePlayback()	**XFaceApp::TaskHandlerBase**	[inline, protected, virtual]
onRewindPlayback() const	**XFaceApp::TaskHandlerBase**	[inline, protected, virtual]
onStopPlayback()	**XFaceApp::TaskHandlerBase**	[inline, protected, virtual]
onUpLoadANIM(const std::string &)	**XFaceApp::TaskHandlerBase**	[inline, protected, virtual]
onUpLoadFAP(const std::string &)	**XFaceApp::TaskHandlerBase**	[inline, protected, virtual]
onUpLoadPHO(const std::string &, const std::string &, const std::string &)	**XFaceApp::TaskHandlerBase**	[inline, protected, virtual]
onUpLoadScript(const std::string &, const std::string &)	**XFaceApp::TaskHandlerBase**	[inline, protected, virtual]
pause(bool bPause=true)	**XFaceApp::TaskHandlerBase**	[inline]
processTask()	**XFaceApp::TaskHandlerBase**	[virtual]

XFaceApp::XMLUtils Member List

This is the complete list of members for **XFaceApp::XMLUtils**, including all inherited members.

xmlify(const Task &task)	**XFaceApp::XMLUtils**	[static]
xmlify(const Notification &note)	**XFaceApp::XMLUtils**	[static]
xmlToNotification(const std::string &xmlString)	**XFaceApp::XMLUtils**	[static]
xmlToTask(const std::string &xmlString)	**XFaceApp::XMLUtils**	[static]

XMath::LineSegment3 Member List

This is the complete list of members for **XMath::LineSegment3**, including all inherited members.

getEnd() const	**XMath::LineSegment3**	[inline]
getStart() const	**XMath::LineSegment3**	[inline]
setEnd(const Vector3 &e)	**XMath::LineSegment3**	[inline]
setStart(const Vector3 &s)	**XMath::LineSegment3**	[inline]

XMath::Matrix4 Member List

This is the complete list of members for **XMath::Matrix4**, including all inherited members.

getInverseTransform() const	**XMath::Matrix4**	
getMatrix()	**XMath::Matrix4**	[inline]
getTranspose() const	**XMath::Matrix4**	
loadIdentity()	**XMath::Matrix4**	
Matrix4(const Matrix4 &pCopy)	**XMath::Matrix4**	
Matrix4()	**XMath::Matrix4**	
Matrix4(float f00, float f01, float f02, float f03, float f10, float f11, float f12, float f13, float f20, float f21, float f22, float f23, float f30, float f31, float f32, float f33)	**XMath::Matrix4**	
Matrix4(const float mat[16])	**XMath::Matrix4**	
negate(void)	**XMath::Matrix4**	
operator *(const Matrix4 &rhs)	**XMath::Matrix4**	
operator *(const Matrix4d &rhs)	**XMath::Matrix4**	
operator *(const float scalar) const	**XMath::Matrix4**	
operator *(const Vector3 &rhs) const	**XMath::Matrix4**	
operator const float *() const	**XMath::Matrix4**	[inline]
operator float *()	**XMath::Matrix4**	[inline]
operator()(unsigned int i, unsigned int j)	**XMath::Matrix4**	[inline]
operator=(const Matrix4 &rhs)	**XMath::Matrix4**	
operator[](int i)	**XMath::Matrix4**	[inline]
operator[](int i) const	**XMath::Matrix4**	[inline]
~Matrix4()	**XMath::Matrix4**	[inline]

XMath::Quaternion Member List

This is the complete list of members for **XMath::Quaternion**, including all inherited members.

FromRotationMatrix(Matrix4 &mat)	**XMath::Quaternion**	
Identity()	**XMath::Quaternion**	
operator *(const Quaternion &rhs) const	**XMath::Quaternion**	
operator+(const Quaternion &rhs) const	**XMath::Quaternion**	
operator=(const Quaternion &rhs)	**XMath::Quaternion**	
Quaternion(float pitch, float yaw, float roll)	**XMath::Quaternion**	
Quaternion(float ix, float iy, float iz, float iw)	**XMath::Quaternion**	
Quaternion(const AxisAngle &axisAngle)	**XMath::Quaternion**	
Quaternion(const Quaternion &rhs)	**XMath::Quaternion**	
Quaternion()	**XMath::Quaternion**	`[inline]`
ToRotationMatrix() const	**XMath::Quaternion**	
x	**XMath::Quaternion**	

XMath::Ray3 Member List

This is the complete list of members for **XMath::Ray3**, including all inherited members.

getDirection() const	**XMath::Ray3**	[inline]
getOrigin() const	**XMath::Ray3**	[inline]
setDirection(const Vector3 &dir)	**XMath::Ray3**	[inline]
setOrigin(const Vector3 &org)	**XMath::Ray3**	[inline]

XMath::Rectangle2 Member List

This is the complete list of members for **XMath::Rectangle2**, including all inherited members.

getBottomRight() const	**XMath::Rectangle2**	[inline]
getHeight() const	**XMath::Rectangle2**	[inline]
getTopLeft() const	**XMath::Rectangle2**	[inline]
getWidth() const	**XMath::Rectangle2**	[inline]
getX0() const	**XMath::Rectangle2**	[inline]
getX1() const	**XMath::Rectangle2**	[inline]
getY0() const	**XMath::Rectangle2**	[inline]
getY1() const	**XMath::Rectangle2**	[inline]
isPointInRect(float x, float y) const	**XMath::Rectangle2**	[inline]
setX0(float _x0)	**XMath::Rectangle2**	[inline]
setX1(float _x1)	**XMath::Rectangle2**	[inline]
setY0(float _y0)	**XMath::Rectangle2**	[inline]
setY1(float _y1)	**XMath::Rectangle2**	[inline]

XMath::Triangle3 Member List

This is the complete list of members for **XMath::Triangle3**, including all inherited members.

getEdge0() const	**XMath::Triangle3**	[inline]
getEdge1() const	**XMath::Triangle3**	[inline]
getOrigin() const	**XMath::Triangle3**	[inline]
setEdge0(const Vector3 &ed0)	**XMath::Triangle3**	[inline]
setEdge1(const Vector3 &ed1)	**XMath::Triangle3**	[inline]
setOrigin(const Vector3 &org)	**XMath::Triangle3**	[inline]
Triangle3(const Vector3 &p0, const Vector3 &p1, const Vector3 &p2)	**XMath::Triangle3**	[inline]

XMath::Vector3 Member List

This is the complete list of members for **XMath::Vector3**, including all inherited members.

cross(const Vector3 &rhs) const	**XMath::Vector3**	[inline]
dot(const Vector3 &rhs) const	**XMath::Vector3**	[inline]
length() const	**XMath::Vector3**	[inline]
lengthSqr() const	**XMath::Vector3**	[inline]
normalize(float tolerance=1e-06)	**XMath::Vector3**	[inline]
operator *(const float scalar) const	**XMath::Vector3**	[inline]
operator *=(const float scalar)	**XMath::Vector3**	[inline]
operator const float *() const	**XMath::Vector3**	[inline]
operator float *()	**XMath::Vector3**	[inline]
operator!=(const Vector3 &rhs) const	**XMath::Vector3**	[inline]
operator+(const Vector3 &rhs) const	**XMath::Vector3**	[inline]
operator+=(const Vector3 &rhs)	**XMath::Vector3**	[inline]
operator-() const	**XMath::Vector3**	[inline]
operator-(const Vector3 &rhs) const	**XMath::Vector3**	[inline]
operator-=(const Vector3 &rhs)	**XMath::Vector3**	[inline]
operator/(float scalar) const	**XMath::Vector3**	[inline]
operator/=(float scalar)	**XMath::Vector3**	[inline]
operator=(const Vector3 &rhs)	**XMath::Vector3**	[inline]
operator==(const Vector3 &rhs) const	**XMath::Vector3**	[inline]
unitCross(const Vector3 &rhs) const	**XMath::Vector3**	[inline]
Vector3(const Vector3 &pCopy)	**XMath::Vector3**	[inline]
Vector3(const float ix, const float iy, const float iz)	**XMath::Vector3**	[inline]
Vector3()	**XMath::Vector3**	[inline]
x	**XMath::Vector3**	
y	**XMath::Vector3**	
z	**XMath::Vector3**	
~Vector3()	**XMath::Vector3**	[inline]

- a -

- addAOIIndex() : **XFace::FDPItem**
- addDrawable() : **XEngine::RenderList**
- addDrawables() : **XEngine::RenderList**

- m_currentFAP : **XFace::IFapStream**
- m_currentTask : **XFaceApp::TaskHandlerBase**
- m_FPS : **XFace::IFapStream**
- m_Height : **XEngine::ITextureFile**
- m_nBits : **XEngine::ITextureFile**
- m_pData : **XEngine::ITextureFile**
- m_version : **XFace::IFapStream**
- m_Width : **XEngine::ITextureFile**
- x : **XMath::Vector3**, **XMath::Quaternion**
- y : **XMath::Vector3**
- z : **XMath::Vector3**

- QUERY : **XFaceApp::ApplicationBase**
- TaskStatus : **XFaceApp::Notification**
- TEXTURETYPE : **XEngine::ITexture**

- kDiscarded : **XFaceApp::Notification**
- kError : **XFaceApp::Notification**
- kFinished : **XFaceApp::Notification**
- kInProgress : **XFaceApp::Notification**
- kInQueue : **XFaceApp::Notification**
- kQueued : **XFaceApp::Notification**
- kStarted : **XFaceApp::Notification**
- kUnknown : **XFaceApp::Notification**
- TEXTURE2D : **XEngine::ITexture**

Here is a list of all documented class members with links to the class documentation for each member:

- c -

- clearList() : **XEngine::RenderList**
- clearParams() : **XFaceApp::Task**
- clone() : **XEngine::Drawable**
- computeFaceNormal() : **XEngine::Geometry**
- computeVertexNormals() : **XEngine::IndexedFaceSet**
- copyFrom() : **XEngine::IndexedFaceSet**, **XEngine::Geometry**
- createFace() : **XFaceApp::ApplicationBase**
- createFapStream() : **XFaceApp::ApplicationBase**
- createRenderer() : **XFaceApp::ApplicationBase**
- createScriptProcessors() : **XFaceApp::ApplicationBase**
- createSound() : **XFaceApp::ApplicationBase**
- createTimer() : **XFaceApp::ApplicationBase**
- cross() : **XMath::Vector3**

Here is a list of all documented class members with links to the class documentation for each member:

- d -

- destroyAll() : **XEngine::TextureManager**, **XEngine::MeshManager**
- discardPendingTasks() : **XFaceApp::TaskHandlerBase**
- dot() : **XMath::Vector3**
- Drawable() : **XEngine::Drawable**

Here is a list of all documented class members with links to the class documentation for each member:

- e -

- enableTexture() : **XEngine::Drawable**
- error() : **XFaceApp::Notification**

Here is a list of all documented class members with links to the class documentation for each member:

- f -

- findItem() : **XFace::FDP**
- fireNotification() : **XFaceApp::ApplicationBase**
- FromRotationMatrix() : **XMath::Quaternion**

Here is a list of all documented class members with links to the class documentation for each member:

- g -

- getAffects() : **XFace::FDPItem**
- getAOI() : **XFace::FDPItem**
- getAOICount() : **XFace::FDPItem**
- getBottomRight() : **XMath::Rectangle2**
- getCurrentFAP() : **XFace::IFapStream**
- getData() : **XEngine::ITextureFile**
- getDirection() : **XMath::Ray3**
- getEdge0() : **XMath::Triangle3**
- getEdge1() : **XMath::Triangle3**
- getEnd() : **XMath::LineSegment3**
- getENS() : **XFace::FDP**
- getES() : **XFace::FDP**
- getFAPCount() : **XFace::IFapStream**, **XFace::FAPFile**
- getFAPU() : **XFace::FDP**
- getFPS() : **XFace::IFapStream**
- getGlobalAxisAngle() : **XFace::FDP**
- getGlobalRotationAngle() : **XFace::FDP**
- getGlobalRotationAxis() : **XFace::FDP**
- getGlobalTranslation() : **XFace::FDP**
- getHeight() : **XMath::Rectangle2**, **XEngine::ITextureFile**
- getID() : **XFaceApp::Task**
- getIndex() : **XFace::FDPItem**
- getIndexCount() : **XEngine::IndexedFaceSet**
- getIndices() : **XEngine::IndexedFaceSet**
- getInstance() : **XEngine::TextureManager**, **XEngine::MorphController**, **XEngine::MeshManager**
- getInverseTransform() : **XMath::Matrix4**
- getIRISD() : **XFace::FDP**
- getItemCount() : **XFace::FDP**

- getItems() : **XFace::FDP**
- getMakerList() : **XFace::InfluenceCalculatorMaker**
- getMatrix() : **XMath::Matrix4**
- getMesh() : **XEngine::MeshManager**
- getMeshInfo() : **XEngine::ModelFileFactory**
- getMeshName() : **XEngine::Drawable**
- getMNS() : **XFace::FDP**
- getMW() : **XFace::FDP**
- getName() : **XFaceApp::Task**, **XFaceApp::Notification**, **XFace::FDPItem**, **XEngine::NamedObj**
- getNBits() : **XEngine::ITextureFile**
- getNormals() : **XEngine::Geometry**
- getOrigin() : **XMath::Triangle3**, **XMath::Ray3**
- getOwnerID() : **XFaceApp::Task**, **XFaceApp::Notification**
- getParamCount() : **XFaceApp::Task**
- getParameter() : **XFaceApp::Task**
- getRotation() : **XEngine::Transform**
- getStart() : **XMath::LineSegment3**
- getStatus() : **XFaceApp::Notification**
- getTaskID() : **XFaceApp::Notification**
- getTexCoords() : **XEngine::Geometry**
- getTexName() : **XEngine::Drawable**
- getTexture() : **XEngine::TextureManager**
- getTextureID() : **XEngine::ITexture**
- getTextureType() : **XEngine::Texture2D**, **XEngine::ITexture**
- getTopLeft() : **XMath::Rectangle2**
- getTranslation() : **XEngine::Transform**
- getTranspose() : **XMath::Matrix4**
- getVersion() : **XFace::IFapStream**
- getVertexCount() : **XEngine::Geometry**
- getVertices() : **XEngine::Geometry**
- getWidth() : **XMath::Rectangle2**, **XEngine::ITextureFile**
- getX0() : **XMath::Rectangle2**
- getX1() : **XMath::Rectangle2**
- getY0() : **XMath::Rectangle2**
- getY1() : **XMath::Rectangle2**

Here is a list of all documented class members with links to the class documentation for each member:

- h -

- hasControlPoint() : **XFace::FDPItem**

Here is a list of all documented class members with links to the class documentation for each member:

- i -

- Identity() : **XMath::Quaternion**
- InfluenceCalculatorMaker() : **XFace::InfluenceCalculatorMaker**
- init() : **XFaceApp::ApplicationBase**
- insertItem() : **XFace::FDP**
- isEnd() : **XFace::IFapStream**, **XFace::FAPFile**
- isFileLoaded() : **XEngine::ModelFileFactory**
- isOpen() : **XFace::IFapStream**, **XFace::FAPFile**
- isPointInRect() : **XMath::Rectangle2**
- isTask() : **XFaceApp::TaskDictionary**
- isTextureOn() : **XEngine::Drawable**
- ITexture() : **XEngine::ITexture**
- ITextureLoader() : **XEngine::ITextureLoader**

Here is a list of all documented class members with links to the class documentation for each member:

- k -

- kDiscarded : **XFaceApp::Notification**
- kError : **XFaceApp::Notification**
- kFinished : **XFaceApp::Notification**
- kInProgress : **XFaceApp::Notification**
- kInQueue : **XFaceApp::Notification**
- kQueued : **XFaceApp::Notification**
- kStarted : **XFaceApp::Notification**
- kUnknown : **XFaceApp::Notification**

Here is a list of all documented class members with links to the class documentation for each member:

- l -

- length() : **XMath::Vector3**
- lengthSqr() : **XMath::Vector3**
- load() : **XEngine::TgaFile**, **XEngine::TextureManager**, **XEngine::TextureLoaderGL**, **XEngine::ITextureLoader**, **XEngine::ITextureFile**, **XEngine::BmpFile**
- loadIdentity() : **XMath::Matrix4**
- loadModel() : **XEngine::VRML97Loader**, **XEngine::VRML1Loader**, **XEngine::OBJLoader**, **XEngine::IModelLoader**
- loadModelFile() : **XEngine::ModelFileFactory**

Here is a list of all documented class members with links to the class documentation for each member:

- m -

- m_currentFAP : **XFace::IFapStream**
- m_currentTask : **XFaceApp::TaskHandlerBase**
- m_FPS : **XFace::IFapStream**
- m_Height : **XEngine::ITextureFile**
- m_nBits : **XEngine::ITextureFile**
- m_pData : **XEngine::ITextureFile**
- m_version : **XFace::IFapStream**
- m_Width : **XEngine::ITextureFile**
- makeInfluenceCalculator() : **XFace::InfluenceCalculatorMaker**
- Matrix4() : **XMath::Matrix4**
- modifyInfluenceCalculator() : **XFace::FDPItem**
- muteAudio() : **XFaceApp::ApplicationBase**

Here is a list of all documented class members with links to the class documentation for each member:

- n -

- NamedObj() : **XEngine::NamedObj**
- negate() : **XMath::Matrix4**
- newInfluenceCalculator() : **XFace::InfluenceCalculatorMaker**
- newTask() : **XFaceApp::TaskHandlerBase**
- next() : **XFace::IFapStream**, **XFace::FAPFile**
- normalize() : **XMath::Vector3**
- Notification() : **XFaceApp::Notification**

Here is a list of all documented class members with links to the class documentation for each member:

- o -

- onAdvanceFrame() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onLoadANIM() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onLoadFAP() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onLoadFDP() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onLoadPHO() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onLoadWAV() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onRenderFrame() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onResumePlayback() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onRewindPlayback() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onStopPlayback() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onUpLoadANIM() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onUpLoadFAP() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onUpLoadPHO() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onUpLoadScript() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- open() : **XFace::IFapStream**, **XFace::FAPFile**

- operator *() : **XMath::Vector3**, **XMath::Quaternion**, **XMath::Matrix4**
- operator *=() : **XMath::Vector3**
- operator const float *() : **XMath::Vector3**, **XMath::Matrix4**
- operator float *() : **XMath::Vector3**, **XMath::Matrix4**
- operator!=() : **XMath::Vector3**, **XFace::XercesString**
- operator()() : **XMath::Matrix4**
- operator+() : **XMath::Vector3**, **XMath::Quaternion**
- operator+=() : **XMath::Vector3**
- operator-() : **XMath::Vector3**
- operator-=() : **XMath::Vector3**
- operator/() : **XMath::Vector3**
- operator/=() : **XMath::Vector3**
- operator=() : **XMath::Vector3**, **XMath::Quaternion**, **XMath::Matrix4**
- operator==() : **XMath::Vector3**
- operator[]() : **XMath::Matrix4**

Here is a list of all documented class members with links to the class documentation for each member:

- p -

- pause() : **XFaceApp::TaskHandlerBase**
- processTask() : **XFaceApp::TaskHandlerBase**
- pushParameter() : **XFaceApp::Task**

Here is a list of all documented class members with links to the class documentation for each member:

- q -

- Quaternion() : **XMath::Quaternion**
- query() : **XFaceApp::ApplicationBase**
- QUERY : **XFaceApp::ApplicationBase**
- queryParamCount() : **XFaceApp::TaskDictionary**

Here is a list of all documented class members with links to the class documentation for each member:

- r -

- registerMesh() : **XEngine::MeshManager**
- registerTask() : **XFaceApp::TaskDictionary**
- registerTexture() : **XEngine::TextureManager**
- releaseInstance() : **XEngine::TextureManager**, **XEngine::MeshManager**
- removeAOIIndex() : **XFace::FDPItem**
- removeDrawable() : **XEngine::RenderList**
- removeInfluenceCalculator() : **XFace::FDPItem**
- render() : **XEngine::RendererGL**, **XEngine::IRenderer**
- renderBegin() : **XFaceApp::ApplicationBase**
- renderEnd() : **XFaceApp::ApplicationBase**
- renderList() : **XEngine::RenderList**
- resetAOI() : **XFace::FDPItem**
- rewind() : **XFace::IFapStream**, **XFace::FAPFile**

Here is a list of all documented class members with links to the class documentation for each member:

- s -

- save() : **XFace::FAPFile**
- setAffects() : **XFace::FDPItem**
- setControlPoint() : **XFace::FDPItem**
- setDirection() : **XMath::Ray3**
- setEdge0() : **XMath::Triangle3**
- setEdge1() : **XMath::Triangle3**
- setEnd() : **XMath::LineSegment3**
- setENS0() : **XFace::FDP**
- setES0() : **XFace::FDP**
- setFAPU() : **XFace::FDP**
- setGlobalRotation() : **XFace::FDP**
- setGlobalTranslation() : **XFace::FDP**
- setID() : **XFaceApp::Task**
- setIndices() : **XEngine::IndexedFaceSet**
- setIRISD0() : **XFace::FDP**
- setMeshName() : **XEngine::Drawable**
- setMNS0() : **XFace::FDP**
- setMW0() : **XFace::FDP**
- setName() : **XFaceApp::Task**, **XFace::FDPItem**, **XEngine::NamedObj**
- setNormals() : **XEngine::Geometry**
- setOrigin() : **XMath::Triangle3**, **XMath::Ray3**
- setOwnerID() : **XFaceApp::Task**
- setRotation() : **XEngine::Transform**
- setScale() : **XEngine::Transform**
- setStart() : **XMath::LineSegment3**
- setStatus() : **XFaceApp::Notification**
- setTexCoords() : **XEngine::Geometry**
- setTexName() : **XEngine::Drawable**

- setTranslation() : **XEngine::Transform**
- setVertices() : **XEngine::Geometry**, **XEngine::DeformableGeometry**
- setX0() : **XMath::Rectangle2**
- setX1() : **XMath::Rectangle2**
- setY0() : **XMath::Rectangle2**
- setY1() : **XMath::Rectangle2**
- synchronize() : **XFaceApp::ApplicationBase**

Here is a list of all documented class members with links to the class documentation for each member:

- t -

- Task() : **XFaceApp::Task**
- TaskStatus : **XFaceApp::Notification**
- TEXTURE2D : **XEngine::ITexture**
- Texture2D() : **XEngine::Texture2D**
- TextureLoaderGL() : **XEngine::TextureLoaderGL**
- TEXTURETYPE : **XEngine::ITexture**
- ToRotationMatrix() : **XMath::Quaternion**
- Triangle3() : **XMath::Triangle3**

Here is a list of all documented class members with links to the class documentation for each member:

- u -

- unitCross() : **XMath::Vector3**
- unLoad() : **XEngine::TextureManager**, **XEngine::TextureLoaderGL**, **XEngine::ITextureLoader**
- unloadAllFiles() : **XEngine::ModelFileFactory**
- unloadModelFile() : **XEngine::ModelFileFactory**
- unregisterTexture() : **XEngine::TextureManager**
- update() : **XEngine::RenderManager**, **XEngine::DeformableGeometry**

Here is a list of all documented class members with links to the class documentation for each member:

- v -

- Vector3() : **XMath::Vector3**

Here is a list of all documented class members with links to the class documentation for each member:

- x -

- x : **XMath::Vector3**, **XMath::Quaternion**
- xmlify() : **XFaceApp::XMLUtils**
- xmlToNotification() : **XFaceApp::XMLUtils**
- xmlToTask() : **XFaceApp::XMLUtils**

Here is a list of all documented class members with links to the class documentation for each member:

- y -

- y : **XMath::Vector3**
- yield() : **XFaceApp::ApplicationBase**

Here is a list of all documented class members with links to the class documentation for each member:

- z -

- z : **XMath::Vector3**

Here is a list of all documented class members with links to the class documentation for each member:

- ~ -

- ~Drawable() : **XEngine::Drawable**
- ~ITexture() : **XEngine::ITexture**
- ~ITextureLoader() : **XEngine::ITextureLoader**
- ~Matrix4() : **XMath::Matrix4**
- ~NamedObj() : **XEngine::NamedObj**
- ~Texture2D() : **XEngine::Texture2D**
- ~TextureLoaderGL() : **XEngine::TextureLoaderGL**
- ~Vector3() : **XMath::Vector3**

- INFINITY : **Vector3.h**

- c -

- clearList() : **XEngine::RenderList**
- clearParams() : **XFaceApp::Task**
- clone() : **XEngine::Drawable**
- computeFaceNormal() : **XEngine::Geometry**
- computeVertexNormals() : **XEngine::IndexedFaceSet**
- copyFrom() : **XEngine::IndexedFaceSet**, **XEngine::Geometry**
- createFace() : **XFaceApp::ApplicationBase**
- createFapStream() : **XFaceApp::ApplicationBase**
- createRenderer() : **XFaceApp::ApplicationBase**
- createScriptProcessors() : **XFaceApp::ApplicationBase**
- createSound() : **XFaceApp::ApplicationBase**
- createTimer() : **XFaceApp::ApplicationBase**
- cross() : **XMath::Vector3**

- d -

- destroyAll() : **XEngine::TextureManager**, **XEngine::MeshManager**
- discardPendingTasks() : **XFaceApp::TaskHandlerBase**
- dot() : **XMath::Vector3**
- Drawable() : **XEngine::Drawable**

- e -

- enableTexture() : **XEngine::Drawable**
- error() : **XFaceApp::Notification**

- f -

- findItem() : **XFace::FDP**
- fireNotification() : **XFaceApp::ApplicationBase**
- FromRotationMatrix() : **XMath::Quaternion**

- g -

- getAffects() : **XFace::FDPItem**
- getAOI() : **XFace::FDPItem**
- getAOICount() : **XFace::FDPItem**
- getBottomRight() : **XMath::Rectangle2**
- getCurrentFAP() : **XFace::IFapStream**
- getData() : **XEngine::ITextureFile**
- getDirection() : **XMath::Ray3**
- getEdge0() : **XMath::Triangle3**
- getEdge1() : **XMath::Triangle3**
- getEnd() : **XMath::LineSegment3**
- getENS() : **XFace::FDP**
- getES() : **XFace::FDP**
- getFAPCount() : **XFace::IFapStream**, **XFace::FAPFile**
- getFAPU() : **XFace::FDP**
- getFPS() : **XFace::IFapStream**
- getGlobalAxisAngle() : **XFace::FDP**
- getGlobalRotationAngle() : **XFace::FDP**
- getGlobalRotationAxis() : **XFace::FDP**
- getGlobalTranslation() : **XFace::FDP**
- getHeight() : **XMath::Rectangle2**, **XEngine::ITextureFile**
- getID() : **XFaceApp::Task**
- getIndex() : **XFace::FDPItem**
- getIndexCount() : **XEngine::IndexedFaceSet**
- getIndices() : **XEngine::IndexedFaceSet**
- getInstance() : **XEngine::TextureManager**, **XEngine::MorphController**, **XEngine::MeshManager**
- getInverseTransform() : **XMath::Matrix4**
- getIRISD() : **XFace::FDP**
- getItemCount() : **XFace::FDP**
- getItems() : **XFace::FDP**

- getMakerList() : **XFace::InfluenceCalculatorMaker**
- getMatrix() : **XMath::Matrix4**
- getMesh() : **XEngine::MeshManager**
- getMeshInfo() : **XEngine::ModelFileFactory**
- getMeshName() : **XEngine::Drawable**
- getMNS() : **XFace::FDP**
- getMW() : **XFace::FDP**
- getName() : **XFaceApp::Task**, **XFaceApp::Notification**, **XFace::FDPItem**, **XEngine::NamedObj**
- getNBits() : **XEngine::ITextureFile**
- getNormals() : **XEngine::Geometry**
- getOrigin() : **XMath::Triangle3**, **XMath::Ray3**
- getOwnerID() : **XFaceApp::Task**, **XFaceApp::Notification**
- getParamCount() : **XFaceApp::Task**
- getParameter() : **XFaceApp::Task**
- getRotation() : **XEngine::Transform**
- getStart() : **XMath::LineSegment3**
- getStatus() : **XFaceApp::Notification**
- getTaskID() : **XFaceApp::Notification**
- getTexCoords() : **XEngine::Geometry**
- getTexName() : **XEngine::Drawable**
- getTexture() : **XEngine::TextureManager**
- getTextureID() : **XEngine::ITexture**
- getTextureType() : **XEngine::Texture2D**, **XEngine::ITexture**
- getTopLeft() : **XMath::Rectangle2**
- getTranslation() : **XEngine::Transform**
- getTranspose() : **XMath::Matrix4**
- getVersion() : **XFace::IFapStream**
- getVertexCount() : **XEngine::Geometry**
- getVertices() : **XEngine::Geometry**
- getWidth() : **XMath::Rectangle2**, **XEngine::ITextureFile**
- getX0() : **XMath::Rectangle2**
- getX1() : **XMath::Rectangle2**
- getY0() : **XMath::Rectangle2**
- getY1() : **XMath::Rectangle2**

- h -

- hasControlPoint() : **XFace::FDPItem**

- i -

- Identity() : **XMath::Quaternion**
- InfluenceCalculatorMaker() : **XFace::InfluenceCalculatorMaker**
- init() : **XFaceApp::ApplicationBase**
- insertItem() : **XFace::FDP**
- isEnd() : **XFace::IFapStream**, **XFace::FAPFile**
- isFileLoaded() : **XEngine::ModelFileFactory**
- isOpen() : **XFace::IFapStream**, **XFace::FAPFile**
- isPointInRect() : **XMath::Rectangle2**
- isTask() : **XFaceApp::TaskDictionary**
- isTextureOn() : **XEngine::Drawable**
- ITexture() : **XEngine::ITexture**
- ITextureLoader() : **XEngine::ITextureLoader**

- l -

- length() : **XMath::Vector3**
- lengthSqr() : **XMath::Vector3**
- load() : **XEngine::TgaFile**, **XEngine::TextureManager**, **XEngine::TextureLoaderGL**, **XEngine::ITextureLoader**, **XEngine::ITextureFile**, **XEngine::BmpFile**
- loadIdentity() : **XMath::Matrix4**
- loadModel() : **XEngine::VRML97Loader**, **XEngine::VRML1Loader**, **XEngine::OBJLoader**, **XEngine::IModelLoader**
- loadModelFile() : **XEngine::ModelFileFactory**

- m -

- makeInfluenceCalculator() : **XFace::InfluenceCalculatorMaker**
- Matrix4() : **XMath::Matrix4**
- modifyInfluenceCalculator() : **XFace::FDPItem**
- muteAudio() : **XFaceApp::ApplicationBase**

- n -

- NamedObj() : **XEngine::NamedObj**
- negate() : **XMath::Matrix4**
- newInfluenceCalculator() : **XFace::InfluenceCalculatorMaker**
- newTask() : **XFaceApp::TaskHandlerBase**
- next() : **XFace::IFapStream**, **XFace::FAPFile**
- normalize() : **XMath::Vector3**
- Notification() : **XFaceApp::Notification**

- o -

- onAdvanceFrame() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onLoadANIM() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onLoadFAP() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onLoadFDP() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onLoadPHO() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onLoadWAV() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onRenderFrame() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onResumePlayback() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onRewindPlayback() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onStopPlayback() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onUpLoadANIM() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onUpLoadFAP() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onUpLoadPHO() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- onUpLoadScript() : **XFaceApp::TaskHandlerBase**, **XFaceApp::ApplicationBase**
- open() : **XFace::IFapStream**, **XFace::FAPFile**
- operator *() : **XMath::Vector3**, **XMath::Quaternion**, **XMath::Matrix4**

- operator *=() : **XMath::Vector3**
- operator const float *() : **XMath::Vector3**, **XMath::Matrix4**
- operator float *() : **XMath::Vector3**, **XMath::Matrix4**
- operator!=() : **XMath::Vector3**, **XFace::XercesString**
- operator()() : **XMath::Matrix4**
- operator+() : **XMath::Vector3**, **XMath::Quaternion**
- operator+=() : **XMath::Vector3**
- operator-() : **XMath::Vector3**
- operator-=() : **XMath::Vector3**
- operator/() : **XMath::Vector3**
- operator/=() : **XMath::Vector3**
- operator=() : **XMath::Vector3**, **XMath::Quaternion**, **XMath::Matrix4**
- operator==() : **XMath::Vector3**
- operator[]() : **XMath::Matrix4**

- p -

- pause() : **XFaceApp::TaskHandlerBase**
- processTask() : **XFaceApp::TaskHandlerBase**
- pushParameter() : **XFaceApp::Task**

- q -

- Quaternion() : **XMath::Quaternion**
- query() : **XFaceApp::ApplicationBase**
- queryParamCount() : **XFaceApp::TaskDictionary**

- r -

- registerMesh() : **XEngine::MeshManager**
- registerTask() : **XFaceApp::TaskDictionary**
- registerTexture() : **XEngine::TextureManager**
- releaseInstance() : **XEngine::TextureManager**, **XEngine::MeshManager**
- removeAOIIndex() : **XFace::FDPItem**
- removeDrawable() : **XEngine::RenderList**
- removeInfluenceCalculator() : **XFace::FDPItem**
- render() : **XEngine::RendererGL**, **XEngine::IRenderer**
- renderBegin() : **XFaceApp::ApplicationBase**
- renderEnd() : **XFaceApp::ApplicationBase**
- renderList() : **XEngine::RenderList**
- resetAOI() : **XFace::FDPItem**
- rewind() : **XFace::IFapStream**, **XFace::FAPFile**

- s -

- save() : **XFace::FAPFile**
- setAffects() : **XFace::FDPItem**
- setControlPoint() : **XFace::FDPItem**
- setDirection() : **XMath::Ray3**
- setEdge0() : **XMath::Triangle3**
- setEdge1() : **XMath::Triangle3**
- setEnd() : **XMath::LineSegment3**
- setENS0() : **XFace::FDP**
- setES0() : **XFace::FDP**
- setFAPU() : **XFace::FDP**
- setGlobalRotation() : **XFace::FDP**
- setGlobalTranslation() : **XFace::FDP**
- setID() : **XFaceApp::Task**
- setIndices() : **XEngine::IndexedFaceSet**
- setIRISD0() : **XFace::FDP**
- setMeshName() : **XEngine::Drawable**
- setMNS0() : **XFace::FDP**
- setMW0() : **XFace::FDP**
- setName() : **XFaceApp::Task**, **XFace::FDPItem**, **XEngine::NamedObj**
- setNormals() : **XEngine::Geometry**
- setOrigin() : **XMath::Triangle3**, **XMath::Ray3**
- setOwnerID() : **XFaceApp::Task**
- setRotation() : **XEngine::Transform**
- setScale() : **XEngine::Transform**
- setStart() : **XMath::LineSegment3**
- setStatus() : **XFaceApp::Notification**
- setTexCoords() : **XEngine::Geometry**
- setTexName() : **XEngine::Drawable**
- setTranslation() : **XEngine::Transform**

- setVertices() : **XEngine::Geometry**, **XEngine::DeformableGeometry**
- setX0() : **XMath::Rectangle2**
- setX1() : **XMath::Rectangle2**
- setY0() : **XMath::Rectangle2**
- setY1() : **XMath::Rectangle2**
- synchronize() : **XFaceApp::ApplicationBase**

- t -

- Task() : **XFaceApp::Task**
- Texture2D() : **XEngine::Texture2D**
- TextureLoaderGL() : **XEngine::TextureLoaderGL**
- ToRotationMatrix() : **XMath::Quaternion**
- Triangle3() : **XMath::Triangle3**

- u -

- unitCross() : **XMath::Vector3**
- unLoad() : **XEngine::TextureManager**, **XEngine::TextureLoaderGL**, **XEngine::ITextureLoader**
- unloadAllFiles() : **XEngine::ModelFileFactory**
- unloadModelFile() : **XEngine::ModelFileFactory**
- unregisterTexture() : **XEngine::TextureManager**
- update() : **XEngine::RenderManager**, **XEngine::DeformableGeometry**

- v -

- Vector3() : **XMath::Vector3**

- x -

- xmlify() : **XFaceApp::XMLUtils**
- xmlToNotification() : **XFaceApp::XMLUtils**
- xmlToTask() : **XFaceApp::XMLUtils**

- y -

- yield() : **XFaceApp::ApplicationBase**

- ~ -

- ~Drawable() : **XEngine::Drawable**
- ~ITexture() : **XEngine::ITexture**
- ~ITextureLoader() : **XEngine::ITextureLoader**
- ~Matrix4() : **XMath::Matrix4**
- ~NamedObj() : **XEngine::NamedObj**
- ~Texture2D() : **XEngine::Texture2D**
- ~TextureLoaderGL() : **XEngine::TextureLoaderGL**
- ~Vector3() : **XMath::Vector3**
