Farouq Adepetu's Shapes

Generated by Doxygen 1.9.4

Chapter 1

Namespace Index

1.1 Namespace List

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

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

Chapter 4

Namespace Documentation

4.1 FAShapes Namespace Reference

Has classes that are used for creating 3D shapes.

Classes

- class Box

  This is class is used to create a box.
- class Cone

  This is class is used to create a cone.
- class Cylinder

  This is class is used to create a cylinder.
- class Pyramid

  This is class is used to create a pyramid.
- class Sphere

  This is class is used to create a sphere.
- class ThreeDimensionalShape
- class Triangle

  The class stores a pointer to a vertex list and indices to the vertices of the triangle.
- struct Vertex

  Data that describes a vertex.

Functions

- void CreateBox (std::vector< FAShapes::Vertex > &vertices, std::vector< FAShapes::Triangle > &triangles)

  Creates the vertices of a unit box and connects them using triangles.
- void CreateCone (std::vector< FAShapes::Vertex > &vertices, std::vector< FAShapes::Triangle > &triangles, unsigned int numVerticesPerCircle=20, unsigned int numCircles=20)

  Creates the vertices of a unit cone and connects them using triangles.
- void CreateCylinder (std::vector< FAShapes::Vertex > &vertices, std::vector< FAShapes::Triangle > &triangles, unsigned int numVerticesPerCircle=20, unsigned int numCircles=20)

  Creates the vertices of a unit cone and connects them using triangles.
- void CreateSphere (std::vector< FAShapes::Vertex > &vertices, std::vector< FAShapes::Triangle > &triangles, unsigned int numVerticesPerCircle=20, unsigned int numCircles=20)

Creates the vertices of a unit sphere and connects them using triangles.

- void CreatePyramid (std::vector< FAShapes::Vertex > &vertices, std::vector< FAShapes::Triangle > &triangles)

  Creates the vertices of a unit pyramid and connects them using triangles.

- void Quad (unsigned int a, unsigned int b, unsigned int c, unsigned int d, std::vector< Triangle > &triangles, FAShapes::Vertex *vertices)

  Stores the indices of the vertices of the triangles that make up a shape.

4.1.1 Detailed Description

Has classes that are used for creating 3D shapes.

4.1.2 Function Documentation

4.1.2.1 CreateBox()

```
void FAShapes::CreateBox (
            std::vector< FAShapes::Vertex > & vertices,
            std::vector< FAShapes::Triangle > & triangles )
```

Creates the vertices of a unit box and connects them using triangles.

Also computes the normal for each vertex.

4.1.2.2 CreateCone()

```
void FAShapes::CreateCone (
            std::vector< FAShapes::Vertex > & vertices,
            std::vector< FAShapes::Triangle > & triangles,
            unsigned int numVerticesPerCircle = 20,
            unsigned int numCircles = 20 )
```

Creates the vertices of a unit cone and connects them using triangles.

Also computes the normal for each vertex. Uses the UV-method to create the vertices of the cone.

4.1.2.3 CreateCylinder()

```
void FAShapes::CreateCylinder (
            std::vector< FAShapes::Vertex > & vertices,
            std::vector< FAShapes::Triangle > & triangles,
            unsigned int numVerticesPerCircle = 20,
            unsigned int numCircles = 20 )
```

Creates the vertices of a unit cone and connects them using triangles.

Also computes the normal for each vertex./n Uses the UV-method to create the vertices of the cylinder.

4.1.2.4 CreatePyramid()

```
void FAShapes::CreatePyramid (
            std::vector< FAShapes::Vertex > & vertices,
            std::vector< FAShapes::Triangle > & triangles )
```

Creates the vertices of a unit pyramid and connects them using triangles.

Also computes the normal for each vertex.

4.1.2.5 CreateSphere()

```
void FAShapes::CreateSphere (
            std::vector< FAShapes::Vertex > & vertices,
            std::vector< FAShapes::Triangle > & triangles,
            unsigned int numVerticesPerCircle = 20,
            unsigned int numCircles = 20 )
```

Creates the vertices of a unit sphere and connects them using triangles.

Also computes the normal for each vertex./n Uses the UV-method to create the vertices of the sphere.

4.1.2.6 Quad()

```
void FAShapes::Quad (
            unsigned int a,
            unsigned int b,
            unsigned int c,
            unsigned int d,
            std::vector< Triangle > & triangles,
            FAShapes::Vertex * vertices )
```

Stores the indices of the vertices of the triangles that make up a shape.

Chapter 5

Class Documentation

5.1 FAShapes::Box Class Reference

This is class is used to create a box.

`#include "FABox.h"`

Public Member Functions

- Box ()

  Creates a Box object. Call InitializeBox to initialize the box.
- void InitializeBox (float width, float height, float depth, const FAMath::Vector4D position, const FAMath::↩ Quaternion orientation, const FAColor::Color &color)

  Initializes the properties of the box.
- const ThreeDimensionalShape & GetShape () const

  Returns the ThreeDimensionalShape object.
- ThreeDimensionalShape & GetShape ()

  Returns the ThreeDimensionalShape object.
- float GetWidth () const

  Returns the width of the box.
- float GetHeight () const

  Returns the height of the box.
- float GetDepth () const

  Returns the depth of the box.
- void SetWidth (float width)

  Sets the width of the box.
- void SetHeight (float height)

  Sets the height of the box.
- void SetDepth (float depth)

  Sets the depth of the box.
- void UpdateModelMatrix ()

  Updates the boxs model matrix.
- float Volume ()

  Returns the volume of the box.

5.1.1 Detailed Description

This is class is used to create a box.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 Box()

```
FAShapes::Box::Box ( )
```

Creates a Box object. Call InitializeBox to initialize the box.

5.1.3 Member Function Documentation

5.1.3.1 GetDepth()

```
float FAShapes::Box::GetDepth ( ) const
```

Returns the depth of the box.

5.1.3.2 GetHeight()

```
float FAShapes::Box::GetHeight ( ) const
```

Returns the height of the box.

5.1.3.3 GetShape() [1/2]

```
ThreeDimensionalShape & FAShapes::Box::GetShape ( )
```

Returns the ThreeDimensionalShape object.

5.1.3.4 GetShape() [2/2]

```
const ThreeDimensionalShape & FAShapes::Box::GetShape ( ) const
```

Returns the ThreeDimensionalShape object.

5.1.3.5 GetWidth()

```
float FAShapes::Box::GetWidth ( ) const
```

Returns the width of the box.

5.1.3.6 InitializeBox()

```
void FAShapes::Box::InitializeBox (
            float width,
            float height,
            float depth,
            const FAMath::Vector4D position,
            const FAMath::Quaternion orientation,
            const FAColor::Color & color )
```

Initializes the properties of the box.

Parameters

in	*width*	The width of the box.
in	*height*	The height of the box.
in	*depth*	The depth of the box.
in	*position*	The position of the box.
in	*orientation*	The orientation of the box.
in	*color*	The color of the box.

5.1.3.7 SetDepth()

```
void FAShapes::Box::SetDepth (
            float depth )
```

Sets the depth of the box.

5.1.3.8 SetHeight()

```
void FAShapes::Box::SetHeight (
            float height )
```

Sets the height of the box.

5.1.3.9 SetWidth()

```
void FAShapes::Box::SetWidth (
            float width )
```

Sets the width of the box.

5.1.3.10 UpdateModelMatrix()

```
void FAShapes::Box::UpdateModelMatrix ( )
```

Updates the boxs model matrix.

5.1.3.11 Volume()

```
float FAShapes::Box::Volume ( )
```

Returns the volume of the box.

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

- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FABox.h

5.2 FAShapes::Cone Class Reference

This is class is used to create a cone.

```
#include "FACone.h"
```

Public Member Functions

- Cone ()

  Creates a Cone object. Call InitializeCone to initialize the cone.
- void InitializeCone (float radius, float height, const FAMath::Vector4D position, const FAMath::Quaternion orientation, const FAColor::Color &color)

  Initializes the properties of the cone.
- const ThreeDimensionalShape & GetShape () const

  Returns the ThreeDimensionalShape object.
- ThreeDimensionalShape & GetShape ()

  Returns the ThreeDimensionalShape object.
- float GetRadius () const

  Returns the radius of the cone.
- float GetHeight () const

  Returns the height of the cone.
- void SetRadius (float radius)

  Sets the radius of the cone.
- void SetHeight (float height)

  Sets the height of the cone.
- void UpdateModelMatrix ()

  Updates the cones model matrix.
- float Volume ()

  Returns the volume of the cone.

5.2.1 Detailed Description

This is class is used to create a cone.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 Cone()

```
FAShapes::Cone::Cone ( )
```

Creates a Cone object. Call InitializeCone to initialize the cone.

5.2.3 Member Function Documentation

5.2.3.1 GetHeight()

```
float FAShapes::Cone::GetHeight ( ) const
```

Returns the height of the cone.

5.2.3.2 GetRadius()

```
float FAShapes::Cone::GetRadius ( ) const
```

Returns the radius of the cone.

5.2.3.3 GetShape() [1/2]

```
ThreeDimensionalShape & FAShapes::Cone::GetShape ( )
```

Returns the ThreeDimensionalShape object.

5.2.3.4 GetShape() [2/2]

```
const ThreeDimensionalShape & FAShapes::Cone::GetShape ( ) const
```

Returns the ThreeDimensionalShape object.

5.2.3.5 InitializeCone()

```
void FAShapes::Cone::InitializeCone (
            float radius,
            float height,
            const FAMath::Vector4D position,
            const FAMath::Quaternion orientation,
            const FAColor::Color & color )
```

Initializes the properties of the cone.

Parameters

in	*width*	The radius of the cone.
in	*height*	The height of the cone.
in	*position*	The position of the cone.
in	*orientation*	The orientation of the cone.
in	*color*	The color of the cone.

5.2.3.6 SetHeight()

```
void FAShapes::Cone::SetHeight (
            float height )
```

Sets the height of the cone.

5.2.3.7 SetRadius()

```
void FAShapes::Cone::SetRadius (
            float radius )
```

Sets the radius of the cone.

5.2.3.8 UpdateModelMatrix()

```
void FAShapes::Cone::UpdateModelMatrix ( )
```

Updates the cones model matrix.

5.2.3.9 Volume()

```
float FAShapes::Cone::Volume ( )
```

Returns the volume of the cone.

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

- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FACone.h

5.3 FAShapes::Cylinder Class Reference

This is class is used to create a cylinder.

```
#include "FACylinder.h"
```

Public Member Functions

- Cylinder ()

  Creates a Cylinder object. Call InitializeCylinder to initialize the cylinder.
- void InitializeCylinder (float radius, float height, const FAMath::Vector4D position, const FAMath::Quaternion orientation, const FAColor::Color &color)

  Initializes the properties of the cylinder.
- const ThreeDimensionalShape & GetShape () const

  Returns the ThreeDimensionalShape object.
- ThreeDimensionalShape & GetShape ()

  Returns the ThreeDimensionalShape object.
- float GetRadius () const

  Returns the radius of the cylinder.
- float GetHeight () const

  Returns the height of the cylinder.
- void SetRadius (float radius)

  Sets the radius of the cylinder.
- void SetHeight (float height)

  Sets the height of the cylinder.
- void UpdateModelMatrix ()

  Updates the cylinders model matrix.
- float Volume ()

  Returns the volume of the cylinder.

5.3.1 Detailed Description

This is class is used to create a cylinder.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 Cylinder()

```
FAShapes::Cylinder::Cylinder ( )
```

Creates a Cylinder object. Call InitializeCylinder to initialize the cylinder.

5.3.3 Member Function Documentation

5.3.3.1 GetHeight()

```
float FAShapes::Cylinder::GetHeight ( ) const
```

Returns the height of the cylinder.

5.3.3.2 GetRadius()

```
float FAShapes::Cylinder::GetRadius ( ) const
```

Returns the radius of the cylinder.

5.3.3.3 GetShape() [1/2]

```
ThreeDimensionalShape & FAShapes::Cylinder::GetShape ( )
```

Returns the ThreeDimensionalShape object.

5.3.3.4 GetShape() [2/2]

```
const ThreeDimensionalShape & FAShapes::Cylinder::GetShape ( ) const
```

Returns the ThreeDimensionalShape object.

5.3.3.5 InitializeCylinder()

```
void FAShapes::Cylinder::InitializeCylinder (
            float radius,
            float height,
            const FAMath::Vector4D position,
            const FAMath::Quaternion orientation,
            const FAColor::Color & color )
```

Initializes the properties of the cylinder.

Parameters

in	*width*	The radius of the cylinder.
in	*height*	The height of the cylinder.
in	*position*	The position of the cylinder.
in	*orientation*	The orientation of the cylinder.
in	*color*	The color of the cylinder.

5.3.3.6 SetHeight()

```
void FAShapes::Cylinder::SetHeight (
            float height )
```

Sets the height of the cylinder.

5.3.3.7 SetRadius()

```
void FAShapes::Cylinder::SetRadius (
            float radius )
```

Sets the radius of the cylinder.

5.3.3.8 UpdateModelMatrix()

```
void FAShapes::Cylinder::UpdateModelMatrix ( )
```

Updates the cylinders model matrix.

5.3.3.9 Volume()

```
float FAShapes::Cylinder::Volume ( )
```

Returns the volume of the cylinder.

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

- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FACylinder.h

5.4 FAShapes::Pyramid Class Reference

This is class is used to create a pyramid.

```
#include "FAPyramid.h"
```

Public Member Functions

- Pyramid ()

  Creates a Pyramid object. Call InitializePyramid to initialize the pyramid.
- void InitializePyramid (float width, float height, float depth, const FAMath::Vector4D position, const FAMath::Quaternion orientation, const FAColor::Color &color)

  Initializes the properties of the pyramid.
- const ThreeDimensionalShape & GetShape () const

  Returns the ThreeDimensionalShape object.
- ThreeDimensionalShape & GetShape ()

  Returns the ThreeDimensionalShape object.
- float GetWidth () const

  Returns the width of the pyramid.
- float GetHeight () const

  Returns the height of the pyramid.
- float GetDepth () const

  Returns the depth of the pyramid.
- void SetWidth (float width)

  Sets the width of the pyramid.
- void SetHeight (float height)

  Sets the height of the pyramid.
- void SetDepth (float depth)

  Sets the depth of the pyramid.
- void UpdateModelMatrix ()

  Updates the pyramids model matrix.
- float Volume ()

  Returns the volume of the pyramid.

5.4.1 Detailed Description

This is class is used to create a pyramid.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 Pyramid()

```
FAShapes::Pyramid::Pyramid ( )
```

Creates a Pyramid object. Call InitializePyramid to initialize the pyramid.

5.4.3 Member Function Documentation

5.4.3.1 GetDepth()

```
float FAShapes::Pyramid::GetDepth ( ) const
```

Returns the depth of the pyramid.

5.4.3.2 GetHeight()

```
float FAShapes::Pyramid::GetHeight ( ) const
```

Returns the height of the pyramid.

5.4.3.3 GetShape() [1/2]

```
ThreeDimensionalShape & FAShapes::Pyramid::GetShape ( )
```

Returns the ThreeDimensionalShape object.

5.4.3.4 GetShape() [2/2]

```
const ThreeDimensionalShape & FAShapes::Pyramid::GetShape ( ) const
```

Returns the ThreeDimensionalShape object.

5.4.3.5 GetWidth()

```
float FAShapes::Pyramid::GetWidth ( ) const
```

Returns the width of the pyramid.

5.4.3.6 InitializePyramid()

```
void FAShapes::Pyramid::InitializePyramid (
            float width,
            float height,
            float depth,
            const FAMath::Vector4D position,
            const FAMath::Quaternion orientation,
            const FAColor::Color & color )
```

Initializes the properties of the pyramid.

Parameters

in	*width*	The width of the pyramid.
in	*height*	The height of the pyramid.
in	*depth*	The depth of the pyramid.
in	*position*	The position of the pyramid.
in	*orientation*	The orientation of the pyramid.
in	*color*	The color of the pyramid.

5.4.3.7 SetDepth()

```
void FAShapes::Pyramid::SetDepth (
            float depth )
```

Sets the depth of the pyramid.

5.4.3.8 SetHeight()

```
void FAShapes::Pyramid::SetHeight (
            float height )
```

Sets the height of the pyramid.

5.4.3.9 SetWidth()

```
void FAShapes::Pyramid::SetWidth (
            float width )
```

Sets the width of the pyramid.

5.4.3.10 UpdateModelMatrix()

```
void FAShapes::Pyramid::UpdateModelMatrix ( )
```

Updates the pyramids model matrix.

5.4.3.11 Volume()

```
float FAShapes::Pyramid::Volume ( )
```

Returns the volume of the pyramid.

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

- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FAPyramid.h

5.5 FAShapes::Sphere Class Reference

This is class is used to create a sphere.

```
#include "FASphere.h"
```

Public Member Functions

- Sphere ()

  Creates a Sphere object. Call InitializeSphere to initialize the sphere.
- void InitializeSphere (float radius, const FAMath::Vector4D position, const FAMath::Quaternion orientation, const FAColor::Color &color)

  Initializes the properties of the sphere.
- const ThreeDimensionalShape & GetShape () const

  Returns the ThreeDimensionalShape object.
- ThreeDimensionalShape & GetShape ()

  Returns the ThreeDimensionalShape object.
- float GetRadius () const

  Returns the radius of the sphere.
- void SetRadius (float radius)

  Sets the radius of the sphere.
- void UpdateModelMatrix ()

  Updates the spheres model matrix.
- float Volume ()

  Returns the volume of the sphere.

5.5.1 Detailed Description

This is class is used to create a sphere.

5.5.2 Constructor & Destructor Documentation

5.5.2.1 Sphere()

```
FAShapes::Sphere::Sphere ( )
```

Creates a Sphere object. Call InitializeSphere to initialize the sphere.

5.5.3 Member Function Documentation

5.5.3.1 GetRadius()

```
float FAShapes::Sphere::GetRadius ( ) const
```

Returns the radius of the sphere.

5.5.3.2 GetShape() [1/2]

```
ThreeDimensionalShape & FAShapes::Sphere::GetShape ( )
```

Returns the ThreeDimensionalShape object.

5.5.3.3 GetShape() [2/2]

```
const ThreeDimensionalShape & FAShapes::Sphere::GetShape ( ) const
```

Returns the ThreeDimensionalShape object.

5.5.3.4 InitializeSphere()

```
void FAShapes::Sphere::InitializeSphere (
            float radius,
            const FAMath::Vector4D position,
            const FAMath::Quaternion orientation,
            const FAColor::Color & color )
```

Initializes the properties of the sphere.

Parameters

in	*width*	The radius of the sphere.
in	*position*	The position of the sphere.
in	*orientation*	The orientation of the sphere.
in	*color*	The color of the sphere.

5.5.3.5 SetRadius()

```
void FAShapes::Sphere::SetRadius (
            float radius )
```

Sets the radius of the sphere.

5.5.3.6 UpdateModelMatrix()

```
void FAShapes::Sphere::UpdateModelMatrix ( )
```

Updates the spheres model matrix.

5.5.3.7 Volume()

```
float FAShapes::Sphere::Volume ( )
```

Returns the volume of the sphere.

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

- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FASphere.h

5.6 FAShapes::ThreeDimensionalShape Class Reference

Public Member Functions

- ThreeDimensionalShape ()

  Default Constructor. Call InitializeThreeDimensionalShape() to initialize the properties of the 3D shape.
- void InitializeThreeDimensionalShape (const FAMath::Vector4D position, const FAMath::Quaternion orientation, const FAColor::Color &color)

  Initializes the properties of the 3D shape.
- const FAMath::Vector4D & GetPosition () const

  Returns the position of the 3D shape.
- const FAMath::Quaternion & GetOrientation () const

  Returns the orientation of the 3D shape.
- const FAColor::Color & GetColor () const

  Returns the color of the 3D shape.
- const FAMath::Matrix4x4 & GetModelMatrix ()

  Returns the model matrix of the 3D shape.
- const FADrawArguments::DrawArguments & GetDrawArguments () const

  Returns the draw arguments of the 3D shape.

- void SetPosition (const FAMath::Vector4D &position)

  Sets the position of the 3D shape.
- void SetOrientation (const FAMath::Quaternion &orientation)

  Sets the orientation of the 3D shape.
- void SetModelMatrix (const FAMath::Matrix4x4 &m)

  Sets the model matrix of the 3D shape.
- void SetDrawArguments (unsigned int indexCount, unsigned int locationFirstIndex, int indexFirstVertex, unsigned int indexConstantData, const std::wstring &constantBufferKey, unsigned int rootParameterIndex, D3D_PRIMITIVE_TOPOLOGY primitive)

  Sets the draw arguments used to render the 3D shape.
- void UpdateShape (FARender::RenderScene *scene, const void *data, unsigned int size)

  Updates the 3D shape constant data.
- void RenderShape (FARender::RenderScene *scene)

  Renders the 3D shape.

5.6.1 Constructor & Destructor Documentation

5.6.1.1 ThreeDimensionalShape()

```
FAShapes::ThreeDimensionalShape::ThreeDimensionalShape ( )
```

Default Constructor. Call InitializeThreeDimensionalShape() to initialize the properties of the 3D shape.

5.6.2 Member Function Documentation

5.6.2.1 GetColor()

```
const FAColor::Color & FAShapes::ThreeDimensionalShape::GetColor ( ) const
```

Returns the color of the 3D shape.

5.6.2.2 GetDrawArguments()

```
const FADrawArguments::DrawArguments & FAShapes::ThreeDimensionalShape::GetDrawArguments ( )
const
```

Returns the draw arguments of the 3D shape.

5.6.2.3 GetModelMatrix()

```
const FAMath::Matrix4x4 & FAShapes::ThreeDimensionalShape::GetModelMatrix ( )
```

Returns the model matrix of the 3D shape.

5.6.2.4 GetOrientation()

```
const FAMath::Quaternion & FAShapes::ThreeDimensionalShape::GetOrientation ( ) const
```

Returns the orientation of the 3D shape.

5.6.2.5 GetPosition()

```
const FAMath::Vector4D & FAShapes::ThreeDimensionalShape::GetPosition ( ) const
```

Returns the position of the 3D shape.

5.6.2.6 InitializeThreeDimensionalShape()

```
void FAShapes::ThreeDimensionalShape::InitializeThreeDimensionalShape (
            const FAMath::Vector4D position,
            const FAMath::Quaternion orientation,
            const FAColor::Color & color )
```

Initializes the properties of the 3D shape.

5.6.2.7 RenderShape()

```
void FAShapes::ThreeDimensionalShape::RenderShape (
            FARender::RenderScene * scene )
```

Renders the 3D shape.

5.6.2.8 SetDrawArguments()

```
void FAShapes::ThreeDimensionalShape::SetDrawArguments (
            unsigned int indexCount,
            unsigned int locationFirstIndex,
            int indexFirstVertex,
            unsigned int indexConstantData,
            const std::wstring & constantBufferKey,
            unsigned int rootParameterIndex,
            D3D_PRIMITIVE_TOPOLOGY primitive )
```

Sets the draw arguments used to render the 3D shape.

5.6.2.9 SetModelMatrix()

```
void FAShapes::ThreeDimensionalShape::SetModelMatrix (
            const FAMath::Matrix4x4 & m )
```

Sets the model matrix of the 3D shape.

5.6.2.10 SetOrientation()

```
void FAShapes::ThreeDimensionalShape::SetOrientation (
            const FAMath::Quaternion & orientation )
```

Sets the orientation of the 3D shape.

5.6.2.11 SetPosition()

```
void FAShapes::ThreeDimensionalShape::SetPosition (
            const FAMath::Vector4D & position )
```

Sets the position of the 3D shape.

5.6.2.12 UpdateShape()

```
void FAShapes::ThreeDimensionalShape::UpdateShape (
            FARender::RenderScene * scene,
            const void * data,
            unsigned int size )
```

Updates the 3D shape constant data.

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

- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FAThree↩ DimensionalShape.h

5.7 FAShapes::Triangle Class Reference

The class stores a pointer to a vertex list and indices to the vertices of the triangle.

`#include "FATriangle.h"`

Public Member Functions

- Triangle (FAShapes::Vertex *vertexList=nullptr, unsigned int p0Index=0, unsigned int p1Index=0, unsigned int p2Index=0)

  Constructs a triangle.
- const FAShapes::Vertex & GetP0 () const

  Returns a constant reference to the P0 vertex of the triangle.
- const FAShapes::Vertex & GetP1 () const

  Returns a constant reference to the P1 vertex of the triangle.
- const FAShapes::Vertex & GetP2 () const

  Returns a constant reference to the P2 vertex of the triangle.
- unsigned int GetP0Index () const

  Returns the index of where P0 is in the vertex list.
- unsigned int GetP1Index () const

  Returns the index of where P1 is in the vertex list.
- unsigned int GetP2Index () const

  Returns the index of where P2 is in the vertex list.
- FAMath::Vector4D GetNormal () const

  Returns the normal of the triangle.
- FAMath::Vector4D GetCenter () const

  Returns the center of the triangle.
- void SetVertexList (FAShapes::Vertex *vertexList)

  Sets the pointer to a vertex list to the specified pointers.
- void SetP0Index (unsigned int index)

  Sets the P0 index to the specified index.
- void SetP1Index (unsigned int index)

  Sets the P1 index to the specified index.
- void SetP2Index (unsigned int index)

  Sets the P2 index to the specified index.
- void SetTriangleIndices (unsigned int p0Index, unsigned int p1Index, unsigned int p2Index)

  Sets the indices of the vertices that make up the triangle to the specified vertices.
- void SetTriangle (FAShapes::Vertex *vertexList, unsigned int p0Index, unsigned int p1Index, unsigned int p2Index)

  Sets the triangle variables.

5.7.1 Detailed Description

The class stores a pointer to a vertex list and indices to the vertices of the triangle.

5.7.2 Constructor & Destructor Documentation

5.7.2.1 Triangle()

```
FAShapes::Triangle::Triangle (
            FAShapes::Vertex * vertexList = nullptr,
            unsigned int p0Index = 0,
            unsigned int p1Index = 0,
            unsigned int p2Index = 0 )
```

Constructs a triangle.

Parameters

in	*vertexList*	A pointer to a vertex list.
in	*p0Index*	The index of the first point of the triangle.
in	*p1Index*	The index of the second point of the triangle.
in	*p2Index*	The index of the third point of the triangle.

5.7.3 Member Function Documentation

5.7.3.1 GetCenter()

```
FAMath::Vector4D FAShapes::Triangle::GetCenter ( ) const
```

Returns the center of the triangle.

5.7.3.2 GetNormal()

```
FAMath::Vector4D FAShapes::Triangle::GetNormal ( ) const
```

Returns the normal of the triangle.

5.7.3.3 GetP0()

```
const FAShapes::Vertex & FAShapes::Triangle::GetP0 ( ) const
```

Returns a constant reference to the P0 vertex of the triangle.

5.7.3.4 GetP0Index()

```
unsigned int FAShapes::Triangle::GetP0Index ( ) const
```

Returns the index of where P0 is in the vertex list.

5.7.3.5 GetP1()

```
const FAShapes::Vertex & FAShapes::Triangle::GetP1 ( ) const
```

Returns a constant reference to the P1 vertex of the triangle.

5.7.3.6 GetP1Index()

```
unsigned int FAShapes::Triangle::GetP1Index ( ) const
```

Returns the index of where P1 is in the vertex list.

5.7.3.7 GetP2()

```
const FAShapes::Vertex & FAShapes::Triangle::GetP2 ( ) const
```

Returns a constant reference to the P2 vertex of the triangle.

5.7.3.8 GetP2Index()

```
unsigned int FAShapes::Triangle::GetP2Index ( ) const
```

Returns the index of where P2 is in the vertex list.

5.7.3.9 SetP0Index()

```
void FAShapes::Triangle::SetP0Index (
            unsigned int index )
```

Sets the P0 index to the specified *index*.

5.7.3.10 SetP1Index()

```
void FAShapes::Triangle::SetP1Index (
            unsigned int index )
```

Sets the P1 index to the specified *index*.

5.7.3.11 SetP2Index()

```
void FAShapes::Triangle::SetP2Index (
            unsigned int index )
```

Sets the P2 index to the specified *index*.

5.7.3.12 SetTriangle()

```
void FAShapes::Triangle::SetTriangle (
            FAShapes::Vertex * vertexList,
            unsigned int p0Index,
            unsigned int p1Index,
            unsigned int p2Index )
```

Sets the triangle variables.

Parameters

in	*vertexList*	A pointer to a vertex list.
in	*p0Index*	The index of the first point of the triangle.
in	*p1Index*	The index of the second point of the triangle.
in	*p2Index*	The index of the third point of the triangle.

5.7.3.13 SetTriangleIndices()

```
void FAShapes::Triangle::SetTriangleIndices (
            unsigned int p0Index,
            unsigned int p1Index,
            unsigned int p2Index )
```

Sets the indices of the vertices that make up the triangle to the specified vertices.

Parameters

in	*p0Index*	The index of the first point of the triangle.
in	*p1Index*	The index of the second point of the triangle.
in	*p2Index*	The index of the third point of the triangle.

5.7.3.14 SetVertexList()

```
void FAShapes::Triangle::SetVertexList (
            FAShapes::Vertex * vertexList )
```

Sets the pointer to a vertex list to the specified pointers.

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

- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FATriangle.h

5.8 FAShapes::Vertex Struct Reference

Data that describes a vertex.

```
#include "FAVertexStructure.h"
```

Public Attributes

- FAMath::Vector4D **position**
- FAMath::Vector4D **normal**
- FAMath::Vector2D **texCoords**

5.8.1 Detailed Description

Data that describes a vertex.

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

- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FAVertex↩ Structure.h

Chapter 6

File Documentation

6.1 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FABox.h File Reference

File has a Box class under the namespace FAShapes.

```
#include "FAThreeDimensionalShape.h"
```

Classes

- class FAShapes::Box

  This is class is used to create a box.

Namespaces

- namespace FAShapes

  Has classes that are used for creating 3D shapes.

6.1.1 Detailed Description

File has a Box class under the namespace FAShapes.

6.2 FABox.h

Go to the documentation of this file.

```
1 #pragma once
2
7 #include "FAThreeDimensionalShape.h"
8
9 namespace FAShapes
10 {
14     class Box
15     {
16     public:
17
21         Box();
22
32         void InitializeBox(float width, float height, float depth, const FAMath::Vector4D position, const
      FAMath::Quaternion orientation,
33             const FAColor::Color& color);
34
37         const ThreeDimensionalShape& GetShape() const;
38
41         ThreeDimensionalShape& GetShape();
42
45         float GetWidth() const;
46
49         float GetHeight() const;
50
53         float GetDepth() const;
54
57         void SetWidth(float width);
58
61         void SetHeight(float height);
62
65         void SetDepth(float depth);
66
69         void UpdateModelMatrix();
70
73         float Volume();
74
75     private:
76         FAShapes::ThreeDimensionalShape mShape;
77
78         float mWidth;
79         float mHeight;
80         float mDepth;
81     };
82 }
```

6.3 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FACone.h File Reference

File has a Cone class under the namespace FAShapes.

```
#include "FAThreeDimensionalShape.h"
```

Classes

- class FAShapes::Cone

  This is class is used to create a cone.

Namespaces

- namespace FAShapes

  Has classes that are used for creating 3D shapes.

6.3.1 Detailed Description

File has a Cone class under the namespace FAShapes.

6.4 FACone.h

Go to the documentation of this file.

```
1 #pragma once
2
7 #include "FAThreeDimensionalShape.h"
8
9 namespace FAShapes
10 {
14     class Cone
15     {
16     public:
17
21         Cone();
22
31         void InitializeCone(float radius, float height, const FAMath::Vector4D position, const
      FAMath::Quaternion orientation,
32             const FAColor::Color& color);
33
36         const ThreeDimensionalShape& GetShape() const;
37
40         ThreeDimensionalShape& GetShape();
41
44         float GetRadius() const;
45
48         float GetHeight() const;
49
52         void SetRadius(float radius);
53
56         void SetHeight(float height);
57
60         void UpdateModelMatrix();
61
64         float Volume();
65
66     private:
67         FAShapes::ThreeDimensionalShape mShape;
68
69         float mRadius;
70         float mHeight;
71     };
72 }
```

6.5 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FACylinder.h File Reference

File has a Cylinder class under the namespace FAShapes.

```
#include "FAThreeDimensionalShape.h"
```

Classes

- class FAShapes::Cylinder

  This is class is used to create a cylinder.

Namespaces

- namespace FAShapes

  Has classes that are used for creating 3D shapes.

6.5.1 Detailed Description

File has a Cylinder class under the namespace FAShapes.

6.6 FACylinder.h

Go to the documentation of this file.

```
1 #pragma once
2
7 #include "FAThreeDimensionalShape.h"
8
9 namespace FAShapes
10 {
14     class Cylinder
15     {
16     public:
17
21         Cylinder();
22
31         void InitializeCylinder(float radius, float height, const FAMath::Vector4D position, const
       FAMath::Quaternion orientation,
32             const FAColor::Color& color);
33
36         const ThreeDimensionalShape& GetShape() const;
37
40         ThreeDimensionalShape& GetShape();
41
44         float GetRadius() const;
45
48         float GetHeight() const;
49
52         void SetRadius(float radius);
53
56         void SetHeight(float height);
57
60         void UpdateModelMatrix();
61
64         float Volume();
65
66     private:
67         FAShapes::ThreeDimensionalShape mShape;
68
69         float mRadius;
70         float mHeight;
71     };
72 }
```

6.7 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FAPyramid.h File Reference

File has a Pyramid class under the namespace FAShapes.

```
#include "FAThreeDimensionalShape.h"
```

Classes

- class FAShapes::Pyramid

  This is class is used to create a pyramid.

Namespaces

- namespace FAShapes

  Has classes that are used for creating 3D shapes.

6.7.1 Detailed Description

File has a Pyramid class under the namespace FAShapes.

6.8 FAPyramid.h

Go to the documentation of this file.

```
1 #pragma once
2
7 #include "FAThreeDimensionalShape.h"
8
9 namespace FAShapes
10 {
14     class Pyramid
15     {
16     public:
17
21         Pyramid();
22
32         void InitializePyramid(float width, float height, float depth, const FAMath::Vector4D position,
       const FAMath::Quaternion orientation,
33             const FAColor::Color& color);
34
37         const ThreeDimensionalShape& GetShape() const;
38
41         ThreeDimensionalShape& GetShape();
42
45         float GetWidth() const;
46
49         float GetHeight() const;
50
53         float GetDepth() const;
54
57         void SetWidth(float width);
58
61         void SetHeight(float height);
62
65         void SetDepth(float depth);
66
69         void UpdateModelMatrix();
70
73         float Volume();
74
75     private:
76         FAShapes::ThreeDimensionalShape mShape;
77
78         float mWidth;
79         float mHeight;
80         float mDepth;
81     };
82 }
```

6.9 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FAShapesUtility.h File Reference

File has utility functions.

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

Namespaces

- namespace FAShapes

  Has classes that are used for creating 3D shapes.

Functions

- void FAShapes::CreateBox (std::vector< FAShapes::Vertex > &vertices, std::vector< FAShapes::Triangle > &triangles)

  Creates the vertices of a unit box and connects them using triangles.
- void FAShapes::CreateCone (std::vector< FAShapes::Vertex > &vertices, std::vector< FAShapes::Triangle > &triangles, unsigned int numVerticesPerCircle=20, unsigned int numCircles=20)

  Creates the vertices of a unit cone and connects them using triangles.
- void FAShapes::CreateCylinder (std::vector< FAShapes::Vertex > &vertices, std::vector< FAShapes::Triangle > &triangles, unsigned int numVerticesPerCircle=20, unsigned int numCircles=20)

  Creates the vertices of a unit cone and connects them using triangles.
- void FAShapes::CreateSphere (std::vector< FAShapes::Vertex > &vertices, std::vector< FAShapes::Triangle > &triangles, unsigned int numVerticesPerCircle=20, unsigned int numCircles=20)

  Creates the vertices of a unit sphere and connects them using triangles.
- void FAShapes::CreatePyramid (std::vector< FAShapes::Vertex > &vertices, std::vector< FAShapes::Triangle > &triangles)

  Creates the vertices of a unit pyramid and connects them using triangles.

6.9.1 Detailed Description

File has utility functions.

6.10 FAShapesUtility.h

Go to the documentation of this file.

```
1 #pragma once
2
7 #include "FATriangle.h"
8 #include <vector>
9
13 namespace FAShapes
14 {
15
20     void CreateBox(std::vector<FAShapes::Vertex>& vertices, std::vector<FAShapes::Triangle>& triangles);
21
27     void CreateCone(std::vector<FAShapes::Vertex>& vertices, std::vector<FAShapes::Triangle>& triangles,
28         unsigned int numVerticesPerCircle = 20, unsigned int numCircles = 20);
29
35     void CreateCylinder(std::vector<FAShapes::Vertex>& vertices, std::vector<FAShapes::Triangle>&
    triangles,
36         unsigned int numVerticesPerCircle = 20, unsigned int numCircles = 20);
37
43     void CreateSphere(std::vector<FAShapes::Vertex>& vertices, std::vector<FAShapes::Triangle>&
    triangles,
44         unsigned int numVerticesPerCircle = 20, unsigned int numCircles = 20);
45
50     void CreatePyramid(std::vector<FAShapes::Vertex>& vertices, std::vector<FAShapes::Triangle>&
    triangles);
51 }
```

6.11 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FASphere.h File Reference

File has a Sphere class under the namespace FAShapes.

```
#include "FAThreeDimensionalShape.h"
```

Classes

- class FAShapes::Sphere

  This is class is used to create a sphere.

Namespaces

- namespace FAShapes

  Has classes that are used for creating 3D shapes.

6.11.1 Detailed Description

File has a Sphere class under the namespace FAShapes.

6.12 FASphere.h

Go to the documentation of this file.

```
1 #pragma once
2
7 #include "FAThreeDimensionalShape.h"
8
9 namespace FAShapes
10 {
14     class Sphere
15     {
16     public:
17
21         Sphere();
22
30         void InitializeSphere(float radius, const FAMath::Vector4D position, const FAMath::Quaternion
      orientation,
31             const FAColor::Color& color);
32
35         const ThreeDimensionalShape& GetShape() const;
36
39         ThreeDimensionalShape& GetShape();
40
43         float GetRadius() const;
44
47         void SetRadius(float radius);
48
51         void UpdateModelMatrix();
52
55         float Volume();
56
57     private:
58         FAShapes::ThreeDimensionalShape mShape;
59
60         float mRadius;
61     };
62 }
```

6.13 FAThreeDimensionalShape.h

```
1 #pragma once
2
3
4 #include "FAColor.h"
5 #include "FADrawArgumentsStructure.h"
6 #include "FARenderScene.h"
7
8 namespace FAShapes
9 {
10     class ThreeDimensionalShape
11     {
12     public:
```

```
13
17          ThreeDimensionalShape();
18
21          void InitializeThreeDimensionalShape(const FAMath::Vector4D position, const FAMath::Quaternion
    orientation,
22              const FAColor::Color& color);
23
26          const FAMath::Vector4D& GetPosition() const;
27
30          const FAMath::Quaternion& GetOrientation() const;
31
34          const FAColor::Color& GetColor() const;
35
38          const FAMath::Matrix4x4& GetModelMatrix();
39
42          const FADrawArguments::DrawArguments& GetDrawArguments() const;
43
46          void SetPosition(const FAMath::Vector4D& position);
47
50          void SetOrientation(const FAMath::Quaternion& orientation);
51
54          void SetModelMatrix(const FAMath::Matrix4x4& m);
55
58          void SetDrawArguments(unsigned int indexCount, unsigned int locationFirstIndex, int
    indexFirstVertex, unsigned int indexConstantData,
59              const std::wstring& constantBufferKey, unsigned int rootParameterIndex,
    D3D_PRIMITIVE_TOPOLOGY primitive);
60
63          void UpdateShape(FARender::RenderScene* scene, const void* data, unsigned int size);
64
67          void RenderShape(FARender::RenderScene* scene);
68
69      private:
70          FAMath::Vector4D mPosition;
71
72          FAMath::Quaternion mOrientation;
73
74          FAColor::Color mColor;
75
76          FAMath::Matrix4x4 mModelMatrix;
77
78          FADrawArguments::DrawArguments mDrawArguments;
79      };
80  }
```

6.14 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FATriangle.h File Reference

File has a Triangle class under the namespace FAShapes.

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

Classes

- class FAShapes::Triangle

  The class stores a pointer to a vertex list and indices to the vertices of the triangle.

Namespaces

- namespace FAShapes

  Has classes that are used for creating 3D shapes.

Functions

- void FAShapes::Quad (unsigned int a, unsigned int b, unsigned int c, unsigned int d, std::vector< Triangle > &triangles, FAShapes::Vertex *vertices)

  Stores the indices of the vertices of the triangles that make up a shape.

6.14.1 Detailed Description

File has a Triangle class under the namespace FAShapes.

6.15 FATriangle.h

Go to the documentation of this file.

```
1 #pragma once
2
3 #include "FAVertexStructure.h"
4 #include <vector>
5
10 namespace FAShapes
11 {
15     class Triangle
16     {
17     public:
18
26         Triangle(FAShapes::Vertex* vertexList = nullptr, unsigned int p0Index = 0, unsigned int p1Index =
      0, unsigned int p2Index = 0);
27
30         const FAShapes::Vertex& GetP0() const;
31
34         const FAShapes::Vertex& GetP1() const;
35
38         const FAShapes::Vertex& GetP2() const;
39
42         unsigned int GetP0Index() const;
43
46         unsigned int GetP1Index() const;
47
50         unsigned int GetP2Index() const;
51
54         FAMath::Vector4D GetNormal() const;
55
58         FAMath::Vector4D GetCenter() const;
59
62         void SetVertexList(FAShapes::Vertex* vertexList);
63
66         void SetP0Index(unsigned int index);
67
70         void SetP1Index(unsigned int index);
71
74         void SetP2Index(unsigned int index);
75
82         void SetTriangleIndices(unsigned int p0Index, unsigned int p1Index, unsigned int p2Index);
83
91         void SetTriangle(FAShapes::Vertex* vertexList, unsigned int p0Index, unsigned int p1Index,
      unsigned int p2Index);
92
93     private:
94         FAShapes::Vertex* mVertexList; //pointer to a vertex list
95         unsigned int mIndexList[3]; //indices into a vertex list
96     };
97
100     void Quad(unsigned int a, unsigned int b, unsigned int c, unsigned int d, std::vector<Triangle>&
      triangles, FAShapes::Vertex* vertices);
101 }
```

6.16 FAVertexStructure.h

```
1 #pragma once
2
3 #include "FAMathEngine.h"
4
5 namespace FAShapes
6 {
10      struct Vertex
11      {
12          FAMath::Vector4D position;
13          FAMath::Vector4D normal;
14          FAMath::Vector2D texCoords;
15      };
16 }
```

Index