

Farouq Adepetu's Shapes

Generated by Doxygen 1.9.4

1 Namespace Index	1
1.1 Namespace List	1
2 Hierarchical Index	3
2.1 Class Hierarchy	3
3 Class Index	5
3.1 Class List	5
4 File Index	7
4.1 File List	7
5 Namespace Documentation	9
5.1 FAShapes Namespace Reference	9
5.1.1 Detailed Description	9
6 Class Documentation	11
6.1 FAShapes::Box Class Reference	11
6.1.1 Detailed Description	12
6.1.2 Constructor & Destructor Documentation	12
6.1.2.1 Box()	12
6.1.3 Member Function Documentation	12
6.1.3.1 GetDepth()	12
6.1.3.2 GetHeight()	12
6.1.3.3 GetWidth()	13
6.1.3.4 SetDepth()	13
6.1.3.5 SetHeight()	13
6.1.3.6 SetWidth()	13
6.1.3.7 UpdateLocalToWorldMatrix()	13
6.1.3.8 Volume()	14
6.2 FAShapes::Cone Class Reference	14
6.2.1 Detailed Description	14
6.2.2 Constructor & Destructor Documentation	15
6.2.2.1 Cone()	15
6.2.3 Member Function Documentation	15
6.2.3.1 GetHeight()	15
6.2.3.2 GetRadius()	15
6.2.3.3 SetHeight()	16
6.2.3.4 SetRadius()	16
6.2.3.5 UpdateLocalToWorldMatrix()	16
6.2.3.6 Volume()	16
6.3 FAShapes::Cylinder Class Reference	16
6.3.1 Constructor & Destructor Documentation	17
6.3.1.1 Cylinder()	17

6.3.2 Member Function Documentation	18
6.3.2.1 GetHeight()	18
6.3.2.2 GetRadius()	18
6.3.2.3 SetHeight()	18
6.3.2.4 SetRadius()	18
6.3.2.5 UpdateLocalToWorldMatrix()	18
6.3.2.6 Volume()	19
6.4 FAShapes::DrawArguments Struct Reference	19
6.4.1 Detailed Description	19
6.5 FAShapes::Pyramid Class Reference	19
6.5.1 Detailed Description	20
6.5.2 Constructor & Destructor Documentation	20
6.5.2.1 Pyramid()	20
6.5.3 Member Function Documentation	21
6.5.3.1 GetDepth()	21
6.5.3.2 GetHeight()	21
6.5.3.3 GetWidth()	21
6.5.3.4 SetDepth()	21
6.5.3.5 SetHeight()	21
6.5.3.6 SetWidth()	22
6.5.3.7 UpdateLocalToWorldMatrix()	22
6.5.3.8 Volume()	22
6.6 FAShapes::Sphere Class Reference	22
6.6.1 Detailed Description	23
6.6.2 Constructor & Destructor Documentation	23
6.6.2.1 Sphere()	23
6.6.3 Member Function Documentation	23
6.6.3.1 GetRadius()	24
6.6.3.2 SetRadius()	24
6.6.3.3 UpdateLocalToWorldMatrix()	24
6.6.3.4 Volume()	24
6.7 FAShapes::ThreeDimensionalShapeAbstract Struct Reference	24
6.7.1 Detailed Description	26
6.7.2 Constructor & Destructor Documentation	27
6.7.2.1 ThreeDimensionalShapeAbstract()	27
6.7.3 Member Function Documentation	27
6.7.3.1 CreateNormals()	27
6.7.3.2 CreateTriangles()	27
6.7.3.3 CreateVertices()	27
6.7.3.4 GetCenter()	28
6.7.3.5 GetColor()	28
6.7.3.6 GetDrawArguments()	28

6.7.3.7 GetLocalToWorldMatrix()	28
6.7.3.8 GetLocalVertices() [1/2]	28
6.7.3.9 GetLocalVertices() [2/2]	28
6.7.3.10 GetNumTriangles()	29
6.7.3.11 GetNumVertices()	29
6.7.3.12 GetTriangle()	29
6.7.3.13 GetTriangleList() [1/2]	29
6.7.3.14 GetTriangleList() [2/2]	29
6.7.3.15 GetXAxis()	29
6.7.3.16 GetYAxis()	30
6.7.3.17 GetZAxis()	30
6.7.3.18 Quad()	30
6.7.3.19 RotateAxes() [1/3]	30
6.7.3.20 RotateAxes() [2/3]	30
6.7.3.21 RotateAxes() [3/3]	31
6.7.3.22 RotateCenter() [1/3]	31
6.7.3.23 RotateCenter() [2/3]	31
6.7.3.24 RotateCenter() [3/3]	31
6.7.3.25 SetCenter() [1/2]	31
6.7.3.26 SetCenter() [2/2]	32
6.7.3.27 SetColor() [1/2]	32
6.7.3.28 SetColor() [2/2]	32
6.7.3.29 SetDrawArguments() [1/2]	32
6.7.3.30 SetDrawArguments() [2/2]	32
6.7.3.31 SetXAxis()	33
6.7.3.32 SetYAxis()	33
6.7.3.33 SetZAxis()	33
6.7.3.34 TranslateCenter() [1/2]	33
6.7.3.35 TranslateCenter() [2/2]	33
6.7.3.36 UpdateLocalToWorldMatrix()	34
6.7.3.37 Volume()	34
6.8 FAShapes::Triangle Class Reference	34
6.8.1 Detailed Description	35
6.8.2 Constructor & Destructor Documentation	35
6.8.2.1 Triangle()	35
6.8.3 Member Function Documentation	35
6.8.3.1 GetCenter()	35
6.8.3.2 GetNormal()	36
6.8.3.3 GetP0()	36
6.8.3.4 GetP0Index()	36
6.8.3.5 GetP1()	36
6.8.3.6 GetP1Index()	36

6.8.3.7 GetP2()	36
6.8.3.8 GetP2Index()	37
6.8.3.9 SetP0Index()	37
6.8.3.10 SetP1Index()	37
6.8.3.11 SetP2Index()	37
6.8.3.12 SetTriangle()	37
6.8.3.13 SetTriangleIndices()	38
6.8.3.14 SetVertexList()	38
6.9 FAShapes::Vertex Struct Reference	38
6.9.1 Detailed Description	38
7 File Documentation	39
7.1 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FABox.h File Reference	39
7.1.1 Detailed Description	39
7.2 FABox.h	40
7.3 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FACone.h File Reference	40
7.3.1 Detailed Description	41
7.4 FACone.h	41
7.5 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FACylinder.h File Reference	41
7.5.1 Detailed Description	42
7.6 FACylinder.h	42
7.7 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FAPyramid.h File Reference	42
7.7.1 Detailed Description	43
7.8 FAPyramid.h	43
7.9 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FAShapesUtility.h File Reference	43
7.9.1 Detailed Description	44
7.10 FAShapesUtility.h	44
7.11 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FASphere.h File Reference	44
7.11.1 Detailed Description	45
7.12 FASphere.h	45
7.13 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FAThreeDimensional.h File Reference	45
7.13.1 Detailed Description	46
7.14 FAThreeDimensional.h	46
7.15 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FATriangle.h File Reference	48
7.15.1 Detailed Description	48
7.16 FATriangle.h	49

Chapter 1

Namespace Index

1.1 Namespace List

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

FAShapes	Has classes that are used for creating 3D shapes	9
--------------------------	--	-------------------

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

FAShapes::DrawArguments	19
FAShapes::ThreeDimensionalShapeAbstract	24
FAShapes::Box	11
FAShapes::Cone	14
FAShapes::Cylinder	16
FAShapes::Pyramid	19
FAShapes::Sphere	22
FAShapes::Triangle	34
FAShapes::Vertex	38

Chapter 3

Class Index

3.1 Class List

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

FAShapes::Box	
This is class is used to create a box	11
FAShapes::Cone	
This is class is used to create a cone	14
FAShapes::Cylinder	16
FAShapes::DrawArguments	
Data that are used as parameters to draw an object	19
FAShapes::Pyramid	
This is class is used to create a pyramid	19
FAShapes::Sphere	
This is class is used to create a sphere	22
FAShapes::ThreeDimensionalShapeAbstract	
An abstract class for 3D shapes	24
FAShapes::Triangle	
The class stores a pointer to a vertex list and indices to the vertices of the triangle	34
FAShapes::Vertex	
Data that describes a vertex	38

Chapter 4

File Index

4.1 File List

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

C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/ FABox.h	
File has a Box class under the namespace FAShapes	39
C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/ FACone.h	
File has a Cone class under the namespace FAShapes	40
C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/ FACylinder.h	
File has a Cylinder class under the namespace FAShapes	41
C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/ FAPyramid.h	
File has a Pyramid class under the namespace FAShapes	42
C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/ FAShapesUtility.h	
File has structures DrawArguments and Vertex under the namespace FAShapes	43
C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/ FASphere.h	
File has a Sphere class under the namespace FAShapes	44
C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/ FAThreeDimensional.h	
File has the abstract class ThreeDimensionalShapeAbstract under the namespace FAShapes	45
C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/ FATriangle.h	
File has a Triangle class under the namespace FAShapes	48

Chapter 5

Namespace Documentation

5.1 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](#)
- struct [DrawArguments](#)
Data that are used as parameters to draw an object.
- class [Pyramid](#)
This is class is used to create a pyramid.
- class [Sphere](#)
This is class is used to create a sphere.
- struct [ThreeDimensionalShapeAbstract](#)
An abstract class for 3D shapes.
- 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.

5.1.1 Detailed Description

Has classes that are used for creating 3D shapes.

Chapter 6

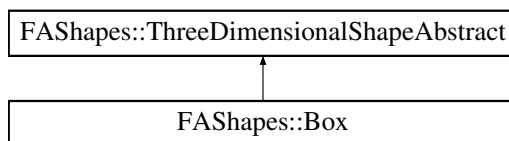
Class Documentation

6.1 FAShapes::Box Class Reference

This class is used to create a box.

```
#include "FABox.h"
```

Inheritance diagram for FAShapes::Box:



Public Member Functions

- **Box** (float width=1.0f, float height=1.0f, float depth=1.0f, const FAShapes::Color &color=FAShapes::Color(0.0f, 0.0f, 0.0f, 1.0f))
*Creates a **Box** with the specified width, height, depth and color centered around the origin.*
- 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 to the specified width.
- void **SetHeight** (float height)
Sets the height of the box to the specified height.
- void **SetDepth** (float depth)
Sets the depth of the box to the specified depth.
- void **UpdateLocalToWorldMatrix** () override final
Updates the box's local to world transformation matrix.
- float **Volume** () override final
Returns the volume of the box.

Additional Inherited Members

6.1.1 Detailed Description

This class is used to create a box.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 Box()

```
FAShapes::Box::Box (
    float width = 1.0f,
    float height = 1.0f,
    float depth = 1.0f,
    const FColor::Color & color = FColor::Color(0.0f, 0.0f, 0.0f, 1.0f) )
```

Creates a [Box](#) with the specified width, height, depth and color centered around the origin.

In a left-handed coordinate system the front face looks towards +z axis, the top face looks towards the +y axis and the right face looks towards the +x axis. The [Box](#) is made using triangles. The vertices are ordered in clockwise order.

Parameters

in	<i>width</i>	The width of the box.
in	<i>height</i>	The height of the box.
in	<i>depth</i>	The depth of the box.
in	<i>color</i>	The color of the box.

6.1.3 Member Function Documentation

6.1.3.1 GetDepth()

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

Returns the depth of the box.

6.1.3.2 GetHeight()

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

Returns the height of the box.

6.1.3.3 GetWidth()

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

Returns the width of the box.

6.1.3.4 SetDepth()

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

Sets the depth of the box to the specified *depth*.

6.1.3.5 SetHeight()

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

Sets the height of the box to the specified *height*.

6.1.3.6 SetWidth()

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

Sets the width of the box to the specified width.

6.1.3.7 UpdateLocalToWorldMatrix()

```
void FAShapes::Box::UpdateLocalToWorldMatrix ( ) [final], [override], [virtual]
```

Updates the boxs local to world transformation matrix.

Implements [FAShapes::ThreeDimensionalShapeAbstract](#).

6.1.3.8 Volume()

```
float FAShapes::Box::Volume ( ) [final], [override], [virtual]
```

Returns the volume of the box.

Implements [FAShapes::ThreeDimensionalShapeAbstract](#).

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

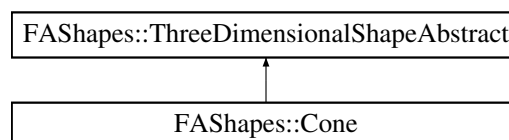
- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/[FABox.h](#)

6.2 FAShapes::Cone Class Reference

This class is used to create a cone.

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

Inheritance diagram for FAShapes::Cone:



Public Member Functions

- [Cone](#) (float radius=1.0f, float height=1.0f, const [FAColor::Color](#) &color=[FAColor::Color](#)(0.0f, 0.0f, 0.0f, 1.0f), unsigned int numCircles=20, unsigned int numVerticesPerCircle=20)
Creates a cone with the specified radius, height and color and it is centered around the origin.
- float [GetRadius](#) () const
Returns the radius of the base of the cone.
- float [GetHeight](#) () const
Returns the height of the base of the cone.
- void [SetRadius](#) (float r)
Sets the radius of the base of the cone to the specified value.
- void [SetHeight](#) (float h)
Sets the height of the base of the cone to the specified value.
- void [UpdateLocalToWorldMatrix](#) () override final
Updates the cones local to world transformation matrix.
- float [Volume](#) () override final
Returns the volume of the cone.

Additional Inherited Members

6.2.1 Detailed Description

This class is used to create a cone.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 Cone()

```
FAShapes::Cone::Cone (
    float radius = 1.0f,
    float height = 1.0f,
    const FAShapes::Color & color = FAShapes::Color(0.0f, 0.0f, 0.0f, 1.0f),
    unsigned int numCircles = 20,
    unsigned int numVerticesPerCircle = 20 )
```

Creates a cone with the specified radius, height and color and it is centered around the origin.

Uses the UV method to create the cone.

The more circles and vertices per circle, the more circular the cone looks.

Parameters

in	<i>radius</i>	The radius of the cone.
in	<i>height</i>	The height of the cone.
in	<i>color</i>	The color of the cone.
in	<i>fillBottom</i>	Pass in true to fill in the bottom of the cone.
in	<i>numCircles</i>	The number of circles the cone has.
in	<i>numVerticesPerCircle</i>	The number of vertices each circle has.

6.2.3 Member Function Documentation

6.2.3.1 GetHeight()

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

Returns the height of the base of the cone.

6.2.3.2 GetRadius()

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

Returns the radius of the base of the cone.

6.2.3.3 SetHeight()

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

Sets the height of the base of the cone to the specified value.

6.2.3.4 SetRadius()

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

Sets the radius of the base of the cone to the specified value.

6.2.3.5 UpdateLocalToWorldMatrix()

```
void FAShapes::Cone::UpdateLocalToWorldMatrix ( ) [final], [override], [virtual]
```

Updates the cones local to world transformation matrix.

Implements [FAShapes::ThreeDimensionalShapeAbstract](#).

6.2.3.6 Volume()

```
float FAShapes::Cone::Volume ( ) [final], [override], [virtual]
```

Returns the volume of the cone.

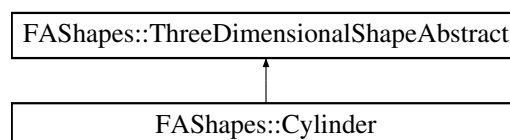
Implements [FAShapes::ThreeDimensionalShapeAbstract](#).

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](#)

6.3 FAShapes::Cylinder Class Reference

Inheritance diagram for FAShapes::Cylinder:



Public Member Functions

- [Cylinder](#) (float radius=1.0f, float height=1.0f, const FAColor::Color &color=FAColor::Color(0.0f, 0.0f, 0.0f, 1.0f), unsigned int numCircles=20, unsigned int numVerticesPerCircle=20)
Creates a cylinder with the specified radius, height and color and it is centered around the origin.
- float [GetRadius](#) () const
Returns the radius of the cylinder.
- float [GetHeight](#) () const
Returns the height of the cylinder.
- void [SetRadius](#) (float r)
Sets the radius of the cylinder to the specified value.
- void [SetHeight](#) (float h)
Sets the height of the cylinder to the specified value.
- void [UpdateLocalToWorldMatrix](#) () override final
Updates the cylinders local to world transformation matrix.
- float [Volume](#) () override final
Returns the volume of the cylinder.

Additional Inherited Members

6.3.1 Constructor & Destructor Documentation

6.3.1.1 Cylinder()

```
FAShapes::Cylinder::Cylinder (
    float radius = 1.0f,
    float height = 1.0f,
    const FAColor::Color & color = FAColor::Color(0.0f, 0.0f, 0.0f, 1.0f),
    unsigned int numCircles = 20,
    unsigned int numVerticesPerCircle = 20 )
```

Creates a cylinder with the specified radius, height and color and it is centered around the origin.

Uses the UV method to create the cylinder.

The more circles and vertices per circle, the more circular it looks.

Parameters

in	<i>radius</i>	The radius of the cylinder.
in	<i>height</i>	The height of the cylinder.
in	<i>color</i>	The color of the cylinder.
in	<i>fillTopAndBottom</i>	Pass in true to fill in the top and bottom of the cylinder.
in	<i>numCircles</i>	The number of circles the cylinder has.
in	<i>numVerticesPerCircle</i>	The number of vertices each circle has.

6.3.2 Member Function Documentation

6.3.2.1 GetHeight()

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

Returns the height of the cylinder.

6.3.2.2 GetRadius()

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

Returns the radius of the cylinder.

6.3.2.3 SetHeight()

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

Sets the height of the cylinder to the specified value.

6.3.2.4 SetRadius()

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

Sets the radius of the cylinder to the specified value.

6.3.2.5 UpdateLocalToWorldMatrix()

```
void FAShapes::Cylinder::UpdateLocalToWorldMatrix ( ) [final], [override], [virtual]
```

Updates the cylinders local to world transformation matrix.

Implements [FAShapes::ThreeDimensionalShapeAbstract](#).

6.3.2.6 Volume()

```
float FAShapes::Cylinder::Volume ( ) [final], [override], [virtual]
```

Returns the volume of the cylinder.

Implements [FAShapes::ThreeDimensionalShapeAbstract](#).

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](#)

6.4 FAShapes::DrawArguments Struct Reference

Data that are used as parameters to draw an object.

```
#include "FAShapesUtility.h"
```

Public Attributes

- unsigned int **indexCount**
- unsigned int **locationOfFirstIndex**
- int **indexOfFirstVertex**
- unsigned int **indexOfConstantData**

6.4.1 Detailed Description

Data that are used as parameters to draw an object.

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

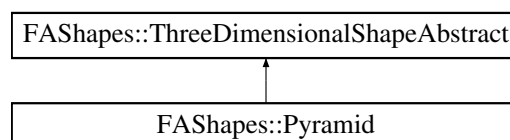
- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/[FAShapesUtility.h](#)

6.5 FAShapes::Pyramid Class Reference

This is class is used to create a pyramid.

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

Inheritance diagram for FAShapes::Pyramid:



Public Member Functions

- [Pyramid](#) (float width=1.0f, float height=1.0f, float depth=1.0f, const FColor::Color &color=FColor::Color(0.0f, 0.0f, 0.0f, 1.0f))
Creates a pyramid with the specified width, height, depth and color centered around the origin.
- 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 to the specified width.
- void [SetHeight](#) (float height)
Sets the height of the pyramid to the specified height.
- void [SetDepth](#) (float depth)
Sets the depth of the pyramid to the specified depth.
- void [UpdateLocalToWorldMatrix](#) () override final
Updates the pyramids local to world transformation matrix.
- float [Volume](#) () override final
Returns the volume of the pyramid.

Additional Inherited Members

6.5.1 Detailed Description

This class is used to create a pyramid.

6.5.2 Constructor & Destructor Documentation

6.5.2.1 Pyramid()

```
FAShapes::Pyramid::Pyramid (
    float width = 1.0f,
    float height = 1.0f,
    float depth = 1.0f,
    const FColor::Color & color = FColor::Color(0.0f, 0.0f, 0.0f, 1.0f) )
```

Creates a pyramid with the specified width, height, depth and color centered around the origin.

In a left-handed coordinate system the front of the pyramid looks towards +z axis, the base of the pyramid looks towards the -y axis and the right face looks towards the +x axis. /n The vertices are ordered in clockwise order.

Parameters

in	<i>width</i>	The width of the pyramid.
in	<i>height</i>	The height of the pyramid.
in	<i>depth</i>	The depth of the pyramid.
in	<i>color</i>	The color of the pyramid.

6.5.3 Member Function Documentation

6.5.3.1 GetDepth()

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

Returns the depth of the pyramid.

6.5.3.2 GetHeight()

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

Returns the height of the pyramid.

6.5.3.3 GetWidth()

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

Returns the width of the pyramid.

6.5.3.4 SetDepth()

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

Sets the depth of the pyramid to the specified *depth*.

6.5.3.5 SetHeight()

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

Sets the height of the pyramid to the specified *height*.

6.5.3.6 SetWidth()

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

Sets the width of the pyramid to the specified *width*.

6.5.3.7 UpdateLocalToWorldMatrix()

```
void FAShapes::Pyramid::UpdateLocalToWorldMatrix ( ) [final], [override], [virtual]
```

Updates the pyramids local to world transformation matrix.

Implements [FAShapes::ThreeDimensionalShapeAbstract](#).

6.5.3.8 Volume()

```
float FAShapes::Pyramid::Volume ( ) [final], [override], [virtual]
```

Returns the volume of the pyramid.

Implements [FAShapes::ThreeDimensionalShapeAbstract](#).

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

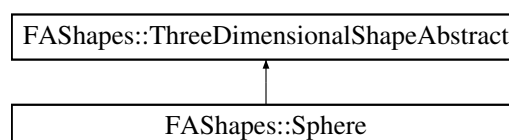
- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/[FAPyramid.h](#)

6.6 FAShapes::Sphere Class Reference

This class is used to create a sphere.

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

Inheritance diagram for FAShapes::Sphere:



Public Member Functions

- [Sphere](#) (float radius=1.0f, const FAColor::Color &color=FAColor::Color(0.0f, 0.0f, 0.0f, 1.0f), unsigned int numCircles=20, unsigned int numVerticesPerCircle=20)
Creates a sphere with the specified radius and color and it is centered around the origin.
- float [GetRadius](#) () const
Returns the radius of the sphere.
- void [SetRadius](#) (float r)
Set the radius of the sphere to the specified value.
- void [UpdateLocalToWorldMatrix](#) () override final
Updates the spheres local to world transformation matrix.
- float [Volume](#) () override final
Returns the volume of the sphere.

Additional Inherited Members

6.6.1 Detailed Description

This is class is used to create a sphere.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 Sphere()

```
FAShapes::Sphere::Sphere (
    float radius = 1.0f,
    const FAColor::Color & color = FAColor::Color(0.0f, 0.0f, 0.0f, 1.0f),
    unsigned int numCircles = 20,
    unsigned int numVerticesPerCircle = 20 )
```

Creates a sphere with the specified radius and color and it is centered around the origin.

Uses the UV method to create the sphere.

The more circles and vertices per circle, the more circular the sphere looks.

Parameters

in	<i>radius</i>	The radius of the cone.
in	<i>color</i>	The color of the cone.
in	<i>numCircles</i>	The number of circles the cone has.
in	<i>numVerticesPerCircle</i>	The number of vertices each circle has.

6.6.3 Member Function Documentation

6.6.3.1 GetRadius()

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

Returns the radius of the sphere.

6.6.3.2 SetRadius()

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

Set the radius of the sphere to the specified value.

6.6.3.3 UpdateLocalToWorldMatrix()

```
void FAShapes::Sphere::UpdateLocalToWorldMatrix ( ) [final], [override], [virtual]
```

Updates the spheres local to world transformation matrix.

Implements [FAShapes::ThreeDimensionalShapeAbstract](#).

6.6.3.4 Volume()

```
float FAShapes::Sphere::Volume ( ) [final], [override], [virtual]
```

Returns the volume of the sphere.

Implements [FAShapes::ThreeDimensionalShapeAbstract](#).

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](#)

6.7 FAShapes::ThreeDimensionalShapeAbstract Struct Reference

An abstract class for 3D shapes.

```
#include "FAThreeDimensional.h"
```

Inheritance diagram for FAShapes::ThreeDimensionalShapeAbstract:



Public Member Functions

- [ThreeDimensionalShapeAbstract](#) (const FAShapes::Color &color)
Constructs a 3D shape.
- const FAShapes::Vector4D & [GetCenter](#) () const
Returns a constant reference to the center of the 3D shape.
- const FAShapes::Vector4D & [GetXAxis](#) () const
Returns a constant reference to the x axis of the 3D shape.
- const FAShapes::Vector4D & [GetYAxis](#) () const
Returns a constant reference to the y axis of the 3D shape.
- const FAShapes::Vector4D & [GetZAxis](#) () const
Returns a constant reference to the z axis of the 3D shape.
- const FAShapes::Matrix4x4 & [GetLocalToWorldMatrix](#) () const
Returns a constant reference to the local to world matrix of the 3D shape.
- const [Vertex](#) * [GetLocalVertices](#) () const
Returns a constant pointer to the local vertices of the 3D shape.
- const [Triangle](#) * [GetTriangleList](#) () const
Returns a constant pointer to the triangles of the 3D shape.
- [Vertex](#) * [GetLocalVertices](#) ()
Returns a pointer to the local vertices of the 3D shape.
- [Triangle](#) * [GetTriangleList](#) ()
Returns a pointer to the triangles of the 3D shape.
- const [Triangle](#) & [GetTriangle](#) (unsigned int index) const
Returns a constant reference to the specified triangle.
- const [DrawArguments](#) & [GetDrawArguments](#) () const
Returns a constant reference to the draw arguments of the 3D shape.
- const FAShapes::Color & [GetColor](#) () const
Returns a constant reference to the color of the 3D shape.
- size_t [GetNumTriangles](#) () const
Returns the number of triangles the 3D shape has.
- size_t [GetNumVertices](#) () const
Returns the number of vertices the 3D shape has.
- void [SetCenter](#) (const FAShapes::Vector4D ¢er)
Sets the center of the 3D shape to the specified vector center.
- void [SetCenter](#) (float x, float y, float z)
Sets the center of the 3D shape to the specified values.
- void [SetXAxis](#) (float x, float y, float z)
Sets the local x-axis of the 3D shape to the specified values.
- void [SetYAxis](#) (float x, float y, float z)
Sets the local y-axis of the 3D shape to the specified values.
- void [SetZAxis](#) (float x, float y, float z)
Sets the local z-axis of the 3D shape to the specified values.
- void [SetColor](#) (const FAShapes::Color &color)
Sets the color of the sphere to the specified color.
- void [SetColor](#) (float r, float g, float b, float a)
Sets the color of the 3D shape to the specified RGBA values.
- void [SetDrawArguments](#) (const [DrawArguments](#) &drawArgs)
Sets the draw arguments of the 3D shape to the specified draw arguments sphereDrawArgs.
- void [SetDrawArguments](#) (unsigned int indexCount, unsigned int locationOfFirstIndex, int indexOfFirstVertex, unsigned int indexOfConstantData)
Sets the draw arguments of the 3D shape to the specified draw arguments.

- void [RotateAxes](#) (const FAMath::Matrix4x4 &rot)
Rotates the local axis of the 3D shape by the specified rotation matrix rot.
- void [RotateAxes](#) (const FAMath::Quaternion &rotQuaternion)
Rotates the local axis of the 3D shape by the specified rotation quaternion rotQuaternion.
- void [RotateAxes](#) (float angle, const FAMath::Vector3D &axis)
Rotates the local axis of the 3D shape by the specified angle around the specified axis.
- void [RotateCenter](#) (const FAMath::Matrix4x4 &rot)
Rotates the center of the 3D shape by the specified rotation matrix rot.
- void [RotateCenter](#) (const FAMath::Quaternion &rotQuaternion)
Rotates the center of the 3D shape by the specified rotation quaternion rotQuaternion.
- void [RotateCenter](#) (float angle, const FAMath::Vector3D &axis)
Rotates the center of the 3D shape by the specified angle around the specified axis.
- void [TranslateCenter](#) (float x, float y, float z)
Translates the center by the specified values.
- void [TranslateCenter](#) (const FAMath::Vector3D &v)
Translates the center by the specified vector v.
- virtual void [UpdateLocalToWorldMatrix](#) ()=0
Updates the local to world matrix for the 3D shape.
- virtual float [Volume](#) ()=0
Returns the volume of the 3D shape.

Protected Member Functions

- void [Quad](#) (unsigned int a, unsigned int b, unsigned int c, unsigned int d)
Stores the indices of the vertices of the triangles that make up the 3D shape.
- virtual void [CreateVertices](#) ()=0
Creates the local vertices of the 3D shape.
- virtual void [CreateTriangles](#) ()=0
Creates the triangles that make up the 3D shape.
- virtual void [CreateNormals](#) ()
Creates the normals of each vertex.

Protected Attributes

- FAMath::Vector4D **mCenter**
- FAMath::Vector4D **mX**
- FAMath::Vector4D **mY**
- FAMath::Vector4D **mZ**
- FAColor::Color **mColor**
- bool **mUpdateLocalToWorldMatrix**
- FAMath::Matrix4x4 **mLocalToWorld**
- std::vector< [Vertex](#) > **mLocalVertices**
- std::vector< [Triangle](#) > **mTriangles**
- [DrawArguments](#) **mSphereDrawArguments** {}

6.7.1 Detailed Description

An abstract class for 3D shapes.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 ThreeDimensionalShapeAbstract()

```
FAShapes::ThreeDimensionalShapeAbstract::ThreeDimensionalShapeAbstract (
    const FAColor::Color & color )
```

Constructs a 3D shape.

Parameters

<code>in</code>	<code><i>color</i></code>	The color if the 3D shape.
-----------------	---------------------------	----------------------------

6.7.3 Member Function Documentation

6.7.3.1 CreateNormals()

```
virtual void FAShapes::ThreeDimensionalShapeAbstract::CreateNormals ( ) [protected], [virtual]
```

Creates the normals of each vertex.

6.7.3.2 CreateTriangles()

```
virtual void FAShapes::ThreeDimensionalShapeAbstract::CreateTriangles ( ) [protected], [pure
virtual]
```

Creates the triangles that make up the 3D shape.

6.7.3.3 CreateVertices()

```
virtual void FAShapes::ThreeDimensionalShapeAbstract::CreateVertices ( ) [protected], [pure
virtual]
```

Creates the local vertices of the 3D shape.

6.7.3.4 GetCenter()

```
const FAMath::Vector4D & FAShapes::ThreeDimensionalShapeAbstract::GetCenter ( ) const
```

Returns a constant reference to the center of the 3D shape.

6.7.3.5 GetColor()

```
const FAShapes::Color & FAShapes::ThreeDimensionalShapeAbstract::GetColor ( ) const
```

Returns a constant reference to the color of the 3D shape.

6.7.3.6 GetDrawArguments()

```
const DrawArguments & FAShapes::ThreeDimensionalShapeAbstract::GetDrawArguments ( ) const
```

Returns a constant reference to the draw arguments of the 3D shape.

6.7.3.7 GetLocalToWorldMatrix()

```
const FAMath::Matrix4x4 & FAShapes::ThreeDimensionalShapeAbstract::GetLocalToWorldMatrix ( ) const
```

Returns a constant reference to the local to world matrix of the 3D shape.

6.7.3.8 GetLocalVertices() [1/2]

```
Vertex * FAShapes::ThreeDimensionalShapeAbstract::GetLocalVertices ( )
```

Returns a pointer to the local vertices of the 3D shape.

6.7.3.9 GetLocalVertices() [2/2]

```
const Vertex * FAShapes::ThreeDimensionalShapeAbstract::GetLocalVertices ( ) const
```

Returns a constant pointer to the local vertices of the 3D shape.

6.7.3.10 GetNumTriangles()

```
size_t FAShapes::ThreeDimensionalShapeAbstract::GetNumTriangles ( ) const
```

Returns the number of triangles the 3D shape has.

6.7.3.11 GetNumVertices()

```
size_t FAShapes::ThreeDimensionalShapeAbstract::GetNumVertices ( ) const
```

Returns the number of vertices the 3D shape has.

6.7.3.12 GetTriangle()

```
const Triangle & FAShapes::ThreeDimensionalShapeAbstract::GetTriangle (
    unsigned int index ) const
```

Returns a constant reference to the specified triangle.

6.7.3.13 GetTriangleList() [1/2]

```
Triangle * FAShapes::ThreeDimensionalShapeAbstract::GetTriangleList ( )
```

Returns a pointer to the triangles of the 3D shape.

6.7.3.14 GetTriangleList() [2/2]

```
const Triangle * FAShapes::ThreeDimensionalShapeAbstract::GetTriangleList ( ) const
```

Returns a constant pointer to the triangles of the 3D shape.

6.7.3.15 GetXAxis()

```
const FAMath::Vector4D & FAShapes::ThreeDimensionalShapeAbstract::GetXAxis ( ) const
```

Returns a constant reference to the x axis of the 3D shape.

6.7.3.16 GetYAxis()

```
const FAMath::Vector4D & FAShapes::ThreeDimensionalShapeAbstract::GetYAxis ( ) const
```

Returns a constant reference to the y axis of the 3D shape.

6.7.3.17 GetZAxis()

```
const FAMath::Vector4D & FAShapes::ThreeDimensionalShapeAbstract::GetZAxis ( ) const
```

Returns a constant reference to the z axis of the 3D shape.

6.7.3.18 Quad()

```
void FAShapes::ThreeDimensionalShapeAbstract::Quad (
    unsigned int a,
    unsigned int b,
    unsigned int c,
    unsigned int d ) [protected]
```

Stores the indices of the vertices of the triangles that make up the 3D shape.

6.7.3.19 RotateAxes() [1/3]

```
void FAShapes::ThreeDimensionalShapeAbstract::RotateAxes (
    const FAMath::Matrix4x4 & rot )
```

Rotates the local axis of the 3D shape by the specified rotation matrix *rot*.

6.7.3.20 RotateAxes() [2/3]

```
void FAShapes::ThreeDimensionalShapeAbstract::RotateAxes (
    const FAMath::Quaternion & rotQuaternion )
```

Rotates the local axis of the 3D shape by the specified rotation quaternion *rotQuaternion*.

6.7.3.21 RotateAxes() [3/3]

```
void FAShapes::ThreeDimensionalShapeAbstract::RotateAxes (
    float angle,
    const FAMath::Vector3D & axis )
```

Rotates the local axis of the 3D shape by the specified *angle* around the specified *axis*.

Uses a quaternion to rotate.

6.7.3.22 RotateCenter() [1/3]

```
void FAShapes::ThreeDimensionalShapeAbstract::RotateCenter (
    const FAMath::Matrix4x4 & rot )
```

Rotates the center of the 3D shape by the specified rotation matrix *rot*.

6.7.3.23 RotateCenter() [2/3]

```
void FAShapes::ThreeDimensionalShapeAbstract::RotateCenter (
    const FAMath::Quaternion & rotQuaternion )
```

Rotates the center of the 3D shape by the specified rotation quaternion *rotQuaternion*.

6.7.3.24 RotateCenter() [3/3]

```
void FAShapes::ThreeDimensionalShapeAbstract::RotateCenter (
    float angle,
    const FAMath::Vector3D & axis )
```

Rotates the center of the 3D shape by the specified *angle* around the specified *axis*.

Uses a quaternion to rotate.

6.7.3.25 SetCenter() [1/2]

```
void FAShapes::ThreeDimensionalShapeAbstract::SetCenter (
    const FAMath::Vector4D & center )
```

Sets the center of the 3D shape to the specified vector *center*.

6.7.3.26 SetCenter() [2/2]

```
void FAShapes::ThreeDimensionalShapeAbstract::SetCenter (
    float x,
    float y,
    float z )
```

Sets the center of the 3D shape to the specified values.

6.7.3.27 SetColor() [1/2]

```
void FAShapes::ThreeDimensionalShapeAbstract::SetColor (
    const FAColor::Color & color )
```

Sets the color of the sphere to the specified *color*.

6.7.3.28 SetColor() [2/2]

```
void FAShapes::ThreeDimensionalShapeAbstract::SetColor (
    float r,
    float g,
    float b,
    float a )
```

Sets the color of the 3D shape to the specified RGBA values.

6.7.3.29 SetDrawArguments() [1/2]

```
void FAShapes::ThreeDimensionalShapeAbstract::SetDrawArguments (
    const DrawArguments & drawArgs )
```

Sets the draw arguments of the 3D shape to the specified draw arguments *sphereDrawArgs*.

6.7.3.30 SetDrawArguments() [2/2]

```
void FAShapes::ThreeDimensionalShapeAbstract::SetDrawArguments (
    unsigned int indexCount,
    unsigned int locationOfFirstIndex,
    int indexOfFirstVertex,
    unsigned int indexOfConstantData )
```

Sets the draw arguments of the 3D shape to the specified draw arguments.

6.7.3.31 SetXAxis()

```
void FAShapes::ThreeDimensionalShapeAbstract::SetXAxis (
    float x,
    float y,
    float z )
```

Sets the local x-axis of the 3D shape to the specified values.

6.7.3.32 SetYAxis()

```
void FAShapes::ThreeDimensionalShapeAbstract::SetYAxis (
    float x,
    float y,
    float z )
```

Sets the local y-axis of the 3D shape to the specified values.

6.7.3.33 SetZAxis()

```
void FAShapes::ThreeDimensionalShapeAbstract::SetZAxis (
    float x,
    float y,
    float z )
```

Sets the local z-axis of the 3D shape to the specified values.

6.7.3.34 TranslateCenter() [1/2]

```
void FAShapes::ThreeDimensionalShapeAbstract::TranslateCenter (
    const FAMath::Vector3D & v )
```

Translates the center by the specified vector *v*.

6.7.3.35 TranslateCenter() [2/2]

```
void FAShapes::ThreeDimensionalShapeAbstract::TranslateCenter (
    float x,
    float y,
    float z )
```

Translates the center by the specified values.

6.7.3.36 UpdateLocalToWorldMatrix()

```
virtual void FAShapes::ThreeDimensionalShapeAbstract::UpdateLocalToWorldMatrix ( ) [pure virtual]
```

Updates the local to world matrix for the 3D shape.

Implemented in [FAShapes::Box](#), [FAShapes::Cone](#), [FAShapes::Cylinder](#), [FAShapes::Pyramid](#), and [FAShapes::Sphere](#).

6.7.3.37 Volume()

```
virtual float FAShapes::ThreeDimensionalShapeAbstract::Volume ( ) [pure virtual]
```

Returns the volume of the 3D shape.

Implemented in [FAShapes::Box](#), [FAShapes::Cone](#), [FAShapes::Cylinder](#), [FAShapes::Pyramid](#), and [FAShapes::Sphere](#).

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/[FAThreeDimensional.h](#)

6.8 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](#) ([Vertex](#) *vertexList=nullptr, unsigned int p0Index=0, unsigned int p1Index=0, unsigned int p2Index=0)
Constructs a triangle.
- const [Vertex](#) & [GetP0](#) () const
Returns a constant reference to the P0 vertex of the triangle.
- const [Vertex](#) & [GetP1](#) () const
Returns a constant reference to the P1 vertex of the triangle.
- const [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](#) ([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](#) ([Vertex](#) *vertexList, unsigned int p0Index, unsigned int p1Index, unsigned int p2Index)
Sets the triangle variables.

6.8.1 Detailed Description

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

6.8.2 Constructor & Destructor Documentation

6.8.2.1 Triangle()

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

Constructs a triangle.

Parameters

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

6.8.3 Member Function Documentation

6.8.3.1 GetCenter()

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

Returns the center of the triangle.

6.8.3.2 GetNormal()

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

Returns the normal of the triangle.

6.8.3.3 GetP0()

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

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

6.8.3.4 GetP0Index()

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

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

6.8.3.5 GetP1()

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

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

6.8.3.6 GetP1Index()

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

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

6.8.3.7 GetP2()

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

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

6.8.3.8 GetP2Index()

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

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

6.8.3.9 SetP0Index()

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

Sets the P0 index to the specified *index*.

6.8.3.10 SetP1Index()

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

Sets the P1 index to the specified *index*.

6.8.3.11 SetP2Index()

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

Sets the P2 index to the specified *index*.

6.8.3.12 SetTriangle()

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

Sets the triangle variables.

Parameters

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

6.8.3.13 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	<i>p0Index</i>	The index of the first point of the triangle.
in	<i>p1Index</i>	The index of the second point of the triangle.
in	<i>p2Index</i>	The index of the third point of the triangle.

6.8.3.14 SetVertexList()

```
void FAShapes::Triangle::SetVertexList (
    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](#)

6.9 FAShapes::Vertex Struct Reference

Data that describes a vertex.

```
#include "FAShapesUtility.h"
```

Public Attributes

- FAMath::Vector4D **position**
- FAColor::Color **color**
- FAMath::Vector4D **normal**
- FAMath::Vector2D **texCoords**

6.9.1 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/[FAShapesUtility.h](#)

Chapter 7

File Documentation

7.1 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 "FAThreeDimensional.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.

7.1.1 Detailed Description

File has a Box class under the namespace [FAShapes](#).

7.2 FBox.h

[Go to the documentation of this file.](#)

```

1  #pragma once
2
3  #include "FAThreeDimensional.h"
4
5  namespace FAShapes
6  {
7      class Box : public ThreeDimensionalShapeAbstract
8      {
9      public:
10
11          Box(float width = 1.0f, float height = 1.0f, float depth = 1.0f,
12              const FAColor::Color& color = FAColor::Color(0.0f, 0.0f, 0.0f, 1.0f));
13
14          float GetWidth() const;
15
16          float GetHeight() const;
17
18          float GetDepth() const;
19
20          void SetWidth(float width);
21
22          void SetHeight(float height);
23
24          void SetDepth(float depth);
25
26          void UpdateLocalToWorldMatrix() override final;
27
28          float Volume() override final;
29
30      private:
31          //Dimensions of the box
32          float mWidth;
33          float mHeight;
34          float mDepth;
35
36          //Creates the vertices of the box.
37          void CreateVertices() override final;
38
39          //Creates the triangles that make up box.
40          void CreateTriangles() override final;
41      };
42  }
```

7.3 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 "FAThreeDimensional.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.

7.3.1 Detailed Description

File has a Cone class under the namespace [FAShapes](#).

7.4 FACone.h

[Go to the documentation of this file.](#)

```

1  #pragma once
2
7  #include "FAThreeDimensional.h"
8
9  namespace FAShapes
10 {
14     class Cone : public ThreeDimensionalShapeAbstract
15     {
16     public:
17
30         Cone(float radius = 1.0f, float height = 1.0f, const FAShapes::Color& color = FAShapes::Color(0.0f,
0.0f, 0.0f, 1.0f),
31             unsigned int numCircles = 20, unsigned int numVerticesPerCircle = 20);
32
35         float GetRadius() const;
36
39         float GetHeight() const;
40
43         void SetRadius(float r);
44
47         void SetHeight(float h);
48
51         void UpdateLocalToWorldMatrix() override final;
52
55         float Volume() override final;
56
57     private:
58
59         //Radius of the cone.
60         float mRadius;
61
62         //Height of the cone
63         float mHeight;
64
65         //The number of slices the cone has.
66         unsigned int mNumCircles;
67
68         //The number of vertices each slice has.
69         unsigned int mNumVerticesPerCircle;
70
71         //Creates the vertices of the cone.
72         void CreateVertices() override final;
73
74         //Creates the triangles that make up the cone.
75         void CreateTriangles() override final;
76
77         void CreateNormals() override final;
78     };
79 }
```

7.5 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FAShapes/Header Files/FACylinder.h File Reference

File has a Cylinder class under the namespace [FAShapes](#).

```
#include "FAThreeDimensional.h"
```

Classes

- class [FAShapes::Cylinder](#)

Namespaces

- namespace [FAShapes](#)

Has classes that are used for creating 3D shapes.

7.5.1 Detailed Description

File has a Cylinder class under the namespace [FAShapes](#).

7.6 FACylinder.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2
3 #include "FAThreeDimensional.h"
4
5 namespace FAShapes
6 {
7     class Cylinder : public ThreeDimensionalShapeAbstract
8     {
9     public:
10
11         Cylinder(float radius = 1.0f, float height = 1.0f, const FAColor::Color& color =
FAColor::Color(0.0f, 0.0f, 0.0f, 1.0f),
12             unsigned int numCircles = 20, unsigned int numVerticesPerCircle = 20);
13
14         float GetRadius() const;
15
16         float GetHeight() const;
17
18         void SetRadius(float r);
19
20         void SetHeight(float h);
21
22         void UpdateLocalToWorldMatrix() override final;
23
24         float Volume() override final;
25
26     private:
27
28         //radius of the cylinder
29         float mRadius;
30
31         //Height of the cylinder
32         float mHeight;
33
34         //The number of slices the cylinder has.
35         unsigned int mNumCircles;
36
37         //The number of vertices each slice has.
38         unsigned int mNumVerticesPerCircle;
39
40         //Creates the vertices of the cylinder.
41         void CreateVertices() override final;
42
43         //Creates the triangles that make up the cylinder.
44         void CreateTriangles() override final;
45
46         void CreateNormals() override final;
47
48     };
49 }
```

7.7 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 "FAThreeDimensional.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.

7.7.1 Detailed Description

File has a Pyramid class under the namespace [FAShapes](#).

7.8 FAPyramid.h

[Go to the documentation of this file.](#)

```

1  #pragma once
2
3  #include "FAThreeDimensional.h"
4
5  namespace FAShapes
6  {
7      class Pyramid : public ThreeDimensionalShapeAbstract
8      {
9      public:
10
11          Pyramid(float width = 1.0f, float height = 1.0f, float depth = 1.0f,
12                  const FAColor::Color& color = FAColor::Color(0.0f, 0.0f, 0.0f, 1.0f));
13
14          float GetWidth() const;
15
16          float GetHeight() const;
17
18          float GetDepth() const;
19
20          void SetWidth(float width);
21
22          void SetHeight(float height);
23
24          void SetDepth(float depth);
25
26          void UpdateLocalToWorldMatrix() override final;
27
28          float Volume() override final;
29
30      private:
31          //Dimensions of the pyramid
32          float mWidth;
33          float mHeight;
34          float mDepth;
35
36          //Creates the vertices of the pyramid.
37          void CreateVertices() override final;
38
39          //Creates the triangles that make up pyramid.
40          void CreateTriangles() override final;
41      };
42  }
```

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

File has structures DrawArguments and Vertex under the namespace [FAShapes](#).

```

#include "FAMathEngine.h"
#include "FAColor.h"
```

Classes

- struct [FAShapes::DrawArguments](#)
Data that are used as parameters to draw an object.
- struct [FAShapes::Vertex](#)
Data that describes a vertex.

Namespaces

- namespace [FAShapes](#)
Has classes that are used for creating 3D shapes.

7.9.1 Detailed Description

File has structures DrawArguments and Vertex under the namespace [FAShapes](#).

7.10 FAShapesUtility.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2
3
4 #include "FAMathEngine.h"
5 #include "FAColor.h"
6
7 namespace FAShapes
8 {
9     struct DrawArguments
10     {
11         unsigned int indexCount;
12         unsigned int locationOfFirstIndex;
13         int indexOfFirstVertex;
14         unsigned int indexOfConstantData;
15     };
16
17     struct Vertex
18     {
19         FAMath::Vector4D position;
20         FAColor::Color color;
21         FAMath::Vector4D normal;
22         FAMath::Vector2D texCoords;
23     };
24 }

```

7.11 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 "FAThreeDimensional.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.

7.11.1 Detailed Description

File has a Sphere class under the namespace [FAShapes](#).

7.12 FASphere.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2
3 #include "FAThreeDimensional.h"
4
5 namespace FAShapes
6 {
7     class Sphere : public ThreeDimensionalShapeAbstract
8     {
9     public:
10
11         Sphere(float radius = 1.0f, const FAColor::Color& color = FAColor::Color(0.0f, 0.0f, 0.0f, 1.0f),
12             unsigned int numCircles = 20, unsigned int numVerticesPerCircle = 20);
13
14         float GetRadius() const;
15
16         void SetRadius(float r);
17
18         void UpdateLocalToWorldMatrix() override final;
19
20         float Volume() override final;
21
22     private:
23         //Radius of the sphere.
24         float mRadius;
25
26         //The number of slices the sphere has.
27         unsigned int mNumCircles;
28
29         //The number of vertices each slice has.
30         unsigned int mNumVerticesPerCircle;
31
32         //Creates the vertices of the sphere.
33         void CreateVertices() override final;
34
35         //Creates the triangles that make up the sphere.
36         void CreateTriangles() override final;
37
38         //Creates the normals of the sphere.
39         void CreateNormals() override final;
40     };
41 }
```

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

File has the abstract class ThreeDimensionalShapeAbstract under the namespace [FAShapes](#).

```

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

Classes

- struct [FAShapes::ThreeDimensionalShapeAbstract](#)
An abstract class for 3D shapes.

Namespaces

- namespace [FAShapes](#)
Has classes that are used for creating 3D shapes.

7.13.1 Detailed Description

File has the abstract class `ThreeDimensionalShapeAbstract` under the namespace [FAShapes](#).

7.14 FAThreeDimensional.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2
3 #include "FATriangle.h"
4 #include <vector>
5
6 namespace FAShapes
7 {
8     class ThreeDimensionalShapeAbstract
9     {
10     public:
11
12         ThreeDimensionalShapeAbstract(const FAColor::Color& color);
13
14         const FAMath::Vector4D& GetCenter() const;
15
16         const FAMath::Vector4D& GetXAxis() const;
17
18         const FAMath::Vector4D& GetYAxis() const;
19
20         const FAMath::Vector4D& GetZAxis() const;
21
22         const FAMath::Matrix4x4& GetLocalToWorldMatrix() const;
23
24         const Vertex* GetLocalVertices() const;
25
26         const Triangle* GetTriangleList() const;
27
28         Vertex* GetLocalVertices();
29
30         Triangle* GetTriangleList();
31
32         const Triangle& GetTriangle(unsigned int index) const;
33
34         const DrawArguments& GetDrawArguments() const;
35
36         const FAColor::Color& GetColor() const;
37
38         size_t GetNumTriangles() const;
39
40         size_t GetNumVertices() const;
41
42         void SetCenter(const FAMath::Vector4D& center);
43
44         void SetCenter(float x, float y, float z);
45
46         void SetXAxis(float x, float y, float z);
47
48         void SetYAxis(float x, float y, float z);
49
50         void SetZAxis(float x, float y, float z);
51
52         void SetColor(const FAColor::Color& color);
53
54     };
55 }
```

```

107     void SetColor(float r, float g, float b, float a);
108
109     void SetDrawArguments(const DrawArguments& drawArgs);
110
111     void SetDrawArguments(unsigned int indexCount, unsigned int locationOfFirstIndex,
112         int indexOffFirstVertex, unsigned int indexOffConstantData);
113
114     void RotateAxes(const FAMath::Matrix4x4& rot);
115
116     void RotateAxes(const FAMath::Quaternion& rotQuaternion);
117
118     void RotateAxes(float angle, const FAMath::Vector3D& axis);
119
120     void RotateCenter(const FAMath::Matrix4x4& rot);
121
122     void RotateCenter(const FAMath::Quaternion& rotQuaternion);
123
124     void RotateCenter(float angle, const FAMath::Vector3D& axis);
125
126     void TranslateCenter(float x, float y, float z);
127
128     void TranslateCenter(const FAMath::Vector3D& v);
129
130     virtual void UpdateLocalToWorldMatrix() = 0;
131
132     virtual float Volume() = 0;
133
134 #if defined(_DEBUG)
135     inline void PrintVertices()
136     {
137         int j = 0;
138         for (auto& i : mLocalVertices)
139         {
140             auto worldPos = i.position * mLocalToWorld;
141             auto worldNormal = i.normal * Transpose(Inverse(mLocalToWorld));
142
143             std::cout << "Vertex " << j << " ";
144             std::cout << std::endl;
145
146             std::cout << "Position: " << "(" << i.position.GetX() << ", " << i.position.GetY() << ", " <<
147 i.position.GetZ()
148             << ", " << i.position.GetW() << ")";
149             std::cout << std::endl;
150
151             std::cout << "Normal: " << "(" << i.normal.GetX() << ", " << i.normal.GetY() << ", " <<
152 i.normal.GetZ()
153             << ", " << i.normal.GetW() << ")";
154             std::cout << std::endl;
155
156             FAMath::Vector4D pos2 = i.position + i.normal;
157             std::cout << "2nd Position: " << "(" << pos2.GetX() << ", " << pos2.GetY() << ", " <<
158 pos2.GetZ()
159             << ", " << pos2.GetW() << ")";
160             std::cout << std::endl;
161
162             std::cout << "Texture Coordinates: " << "(" << i.texCoords.GetX() << ", " <<
163 i.texCoords.GetY() << ")";
164             std::cout << std::endl;
165
166             std::cout << "World Position: " << "(" << worldPos.GetX() << ", " << worldPos.GetY() << ", " <<
167 worldPos.GetZ() << ", " << worldPos.GetW() << ")";
168             std::cout << std::endl;
169
170             std::cout << "World Normal: " << "(" << worldNormal.GetX() << ", " << worldNormal.GetY() << ", " <<
171 worldNormal.GetZ() << ", " << worldNormal.GetW() << ")";
172             std::cout << std::endl;
173
174             std::cout << std::endl;
175             ++j;
176         }
177     }
178 #endif
179
180     protected:
181         //Center of the 3D shape.
182         FAMath::Vector4D mCenter;
183
184         //Local axes of the 3D shape.
185         FAMath::Vector4D mX;
186         FAMath::Vector4D mY;
187         FAMath::Vector4D mZ;
188
189         //Color of the 3D shape.
190         FIColor::Color mColor;

```

```

216
217     //If true, updates the local to world matrix.
218     bool mUpdateLocalToWorldMatrix;
219
220     //Local to world matrix of the 3D shape.
221     FAMath::Matrix4x4 mLocalToWorld;
222
223     //Local vertices of the 3D shape.
224     std::vector<Vertex> mLocalVertices;
225
226     //The triangles that make up the 3D shape.
227     std::vector<Triangle> mTriangles;
228
229     //The arguments needed to render the 3D shape.
230     DrawArguments mSphereDrawArguments{};
231
232     void Quad(unsigned int a, unsigned int b, unsigned int c, unsigned int d);
233
234     virtual void CreateVertices() = 0;
235
236     virtual void CreateTriangles() = 0;
237
238     virtual void CreateNormals();
239 };
240 }

```

7.15 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 "FAShapesUtility.h"
```

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.

7.15.1 Detailed Description

File has a Triangle class under the namespace [FAShapes](#).

7.16 FATriangle.h

[Go to the documentation of this file.](#)

```

1  #pragma once
2
3  #include "FAShapesUtility.h"
4
5  namespace FAShapes
6  {
7      class Triangle
8      {
9      public:
10         Triangle(Vertex* vertexList = nullptr, unsigned int p0Index = 0, unsigned int p1Index = 0,
11         unsigned int p2Index = 0);
12
13         const Vertex& GetP0() const;
14
15         const Vertex& GetP1() const;
16
17         const Vertex& GetP2() const;
18
19         unsigned int GetP0Index() const;
20
21         unsigned int GetP1Index() const;
22
23         unsigned int GetP2Index() const;
24
25         FAMath::Vector4D GetNormal() const;
26
27         FAMath::Vector4D GetCenter() const;
28
29         void SetVertexList(Vertex* vertexList);
30
31         void SetP0Index(unsigned int index);
32
33         void SetP1Index(unsigned int index);
34
35         void SetP2Index(unsigned int index);
36
37         void SetTriangleIndices(unsigned int p0Index, unsigned int p1Index, unsigned int p2Index);
38
39         void SetTriangle(Vertex* vertexList, unsigned int p0Index, unsigned int p1Index, unsigned int
40         p2Index);
41
42     private:
43         Vertex* mVertexList; //pointer to a vertex list
44         unsigned int mIndexList[3]; //indices into a vertex list
45     };
46 }

```


Index

- Box
 - FAShapes::Box, [12](#)
- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FABox.h, [39](#), [40](#)
- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FACone.h, [40](#), [41](#)
- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FACylinder.h, [41](#), [42](#)
- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FAPyramid.h, [42](#), [43](#)
- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FAShapesUtility.h, [43](#), [44](#)
- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FASphere.h, [44](#), [45](#)
- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FAThreeDimensionalShape.h, [45](#), [46](#)
- C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FATriangle.h, [48](#), [49](#)
- Cone
 - FAShapes::Cone, [15](#)
- CreateNormals
 - FAShapes::ThreeDimensionalShapeAbstract, [27](#)
- CreateTriangles
 - FAShapes::ThreeDimensionalShapeAbstract, [27](#)
- CreateVertices
 - FAShapes::ThreeDimensionalShapeAbstract, [27](#)
- Cylinder
 - FAShapes::Cylinder, [17](#)
- FAShapes, [9](#)
- FAShapes::Box, [11](#)
 - Box, [12](#)
 - GetDepth, [12](#)
 - GetHeight, [12](#)
 - GetWidth, [12](#)
 - SetDepth, [13](#)
 - SetHeight, [13](#)
 - SetWidth, [13](#)
 - UpdateLocalToWorldMatrix, [13](#)
 - Volume, [13](#)
- FAShapes::Cone, [14](#)
 - Cone, [15](#)
 - GetHeight, [15](#)
 - GetRadius, [15](#)
 - SetHeight, [15](#)
 - SetRadius, [16](#)
 - UpdateLocalToWorldMatrix, [16](#)
 - Volume, [16](#)
- FAShapes::Cylinder, [16](#)
 - Cylinder, [17](#)
 - GetHeight, [18](#)
 - GetRadius, [18](#)
 - SetHeight, [18](#)
 - SetRadius, [18](#)
 - UpdateLocalToWorldMatrix, [18](#)
 - Volume, [18](#)
- FAShapes::DrawArguments, [19](#)
- FAShapes::Pyramid, [19](#)
 - GetDepth, [21](#)
 - GetHeight, [21](#)
 - GetWidth, [21](#)
 - Pyramid, [20](#)
 - SetDepth, [21](#)
 - SetHeight, [21](#)
 - SetWidth, [21](#)
 - UpdateLocalToWorldMatrix, [22](#)
 - Volume, [22](#)
- FAShapes::Sphere, [22](#)
 - GetRadius, [23](#)
 - SetRadius, [24](#)
 - Sphere, [23](#)
 - UpdateLocalToWorldMatrix, [24](#)
 - Volume, [24](#)
- FAShapes::ThreeDimensionalShapeAbstract, [24](#)
 - CreateNormals, [27](#)
 - CreateTriangles, [27](#)
 - CreateVertices, [27](#)
 - GetCenter, [27](#)
 - GetColor, [28](#)
 - GetDrawArguments, [28](#)
 - GetLocalToWorldMatrix, [28](#)
 - GetLocalVertices, [28](#)
 - GetNumTriangles, [28](#)
 - GetNumVertices, [29](#)
 - GetTriangle, [29](#)
 - GetTriangleList, [29](#)
 - GetXAxis, [29](#)
 - GetYAxis, [29](#)
 - GetZAxis, [30](#)
 - Quad, [30](#)

- RotateAxes, [30](#)
- RotateCenter, [31](#)
- SetCenter, [31](#)
- SetColor, [32](#)
- SetDrawArguments, [32](#)
- SetXAxis, [32](#)
- SetYAxis, [33](#)
- SetZAxis, [33](#)
- ThreeDimensionalShapeAbstract, [27](#)
- TranslateCenter, [33](#)
- UpdateLocalToWorldMatrix, [33](#)
- Volume, [34](#)
- FAShapes::Triangle, [34](#)
 - GetCenter, [35](#)
 - GetNormal, [36](#)
 - GetP0, [36](#)
 - GetP0Index, [36](#)
 - GetP1, [36](#)
 - GetP1Index, [36](#)
 - GetP2, [36](#)
 - GetP2Index, [36](#)
 - SetP0Index, [37](#)
 - SetP1Index, [37](#)
 - SetP2Index, [37](#)
 - SetTriangle, [37](#)
 - SetTriangleIndices, [38](#)
 - SetVertexList, [38](#)
 - Triangle, [35](#)
- FAShapes::Vertex, [38](#)
- GetCenter
 - FAShapes::ThreeDimensionalShapeAbstract, [27](#)
 - FAShapes::Triangle, [35](#)
- GetColor
 - FAShapes::ThreeDimensionalShapeAbstract, [28](#)
- GetDepth
 - FAShapes::Box, [12](#)
 - FAShapes::Pyramid, [21](#)
- GetDrawArguments
 - FAShapes::ThreeDimensionalShapeAbstract, [28](#)
- GetHeight
 - FAShapes::Box, [12](#)
 - FAShapes::Cone, [15](#)
 - FAShapes::Cylinder, [18](#)
 - FAShapes::Pyramid, [21](#)
- GetLocalToWorldMatrix
 - FAShapes::ThreeDimensionalShapeAbstract, [28](#)
- GetLocalVertices
 - FAShapes::ThreeDimensionalShapeAbstract, [28](#)
- GetNormal
 - FAShapes::Triangle, [36](#)
- GetNumTriangles
 - FAShapes::ThreeDimensionalShapeAbstract, [28](#)
- GetNumVertices
 - FAShapes::ThreeDimensionalShapeAbstract, [29](#)
- GetP0
 - FAShapes::Triangle, [36](#)
- GetP0Index
 - FAShapes::Triangle, [36](#)
- GetP1
 - FAShapes::Triangle, [36](#)
- GetP1Index
 - FAShapes::Triangle, [36](#)
- GetP2
 - FAShapes::Triangle, [36](#)
- GetP2Index
 - FAShapes::Triangle, [36](#)
- GetRadius
 - FAShapes::Cone, [15](#)
 - FAShapes::Cylinder, [18](#)
 - FAShapes::Sphere, [23](#)
- GetTriangle
 - FAShapes::ThreeDimensionalShapeAbstract, [29](#)
- GetTriangleList
 - FAShapes::ThreeDimensionalShapeAbstract, [29](#)
- GetWidth
 - FAShapes::Box, [12](#)
 - FAShapes::Pyramid, [21](#)
- GetXAxis
 - FAShapes::ThreeDimensionalShapeAbstract, [29](#)
- GetYAxis
 - FAShapes::ThreeDimensionalShapeAbstract, [29](#)
- GetZAxis
 - FAShapes::ThreeDimensionalShapeAbstract, [30](#)
- Pyramid
 - FAShapes::Pyramid, [20](#)
- Quad
 - FAShapes::ThreeDimensionalShapeAbstract, [30](#)
- RotateAxes
 - FAShapes::ThreeDimensionalShapeAbstract, [30](#)
- RotateCenter
 - FAShapes::ThreeDimensionalShapeAbstract, [31](#)
- SetCenter
 - FAShapes::ThreeDimensionalShapeAbstract, [31](#)
- SetColor
 - FAShapes::ThreeDimensionalShapeAbstract, [32](#)
- SetDepth
 - FAShapes::Box, [13](#)
 - FAShapes::Pyramid, [21](#)
- SetDrawArguments
 - FAShapes::ThreeDimensionalShapeAbstract, [32](#)
- SetHeight
 - FAShapes::Box, [13](#)
 - FAShapes::Cone, [15](#)
 - FAShapes::Cylinder, [18](#)
 - FAShapes::Pyramid, [21](#)
- SetP0Index
 - FAShapes::Triangle, [37](#)
- SetP1Index
 - FAShapes::Triangle, [37](#)
- SetP2Index
 - FAShapes::Triangle, [37](#)
- SetRadius
 - FAShapes::Cone, [16](#)

- FAShapes::Cylinder, [18](#)
 - FAShapes::Sphere, [24](#)
- SetTriangle
 - FAShapes::Triangle, [37](#)
- SetTriangleIndices
 - FAShapes::Triangle, [38](#)
- SetVertexList
 - FAShapes::Triangle, [38](#)
- SetWidth
 - FAShapes::Box, [13](#)
 - FAShapes::Pyramid, [21](#)
- SetXAxis
 - FAShapes::ThreeDimensionalShapeAbstract, [32](#)
- SetYAxis
 - FAShapes::ThreeDimensionalShapeAbstract, [33](#)
- SetZAxis
 - FAShapes::ThreeDimensionalShapeAbstract, [33](#)
- Sphere
 - FAShapes::Sphere, [23](#)
- ThreeDimensionalShapeAbstract
 - FAShapes::ThreeDimensionalShapeAbstract, [27](#)
- TranslateCenter
 - FAShapes::ThreeDimensionalShapeAbstract, [33](#)
- Triangle
 - FAShapes::Triangle, [35](#)
- UpdateLocalToWorldMatrix
 - FAShapes::Box, [13](#)
 - FAShapes::Cone, [16](#)
 - FAShapes::Cylinder, [18](#)
 - FAShapes::Pyramid, [22](#)
 - FAShapes::Sphere, [24](#)
 - FAShapes::ThreeDimensionalShapeAbstract, [33](#)
- Volume
 - FAShapes::Box, [13](#)
 - FAShapes::Cone, [16](#)
 - FAShapes::Cylinder, [18](#)
 - FAShapes::Pyramid, [22](#)
 - FAShapes::Sphere, [24](#)
 - FAShapes::ThreeDimensionalShapeAbstract, [34](#)