

# Farouq Adepetu's Shapes

Generated by Doxygen 1.9.4



<b>1 Namespace Index</b>	<b>1</b>
1.1 Namespace List	1
<b>2 Class Index</b>	<b>3</b>
2.1 Class List	3
<b>3 File Index</b>	<b>5</b>
3.1 File List	5
<b>4 Namespace Documentation</b>	<b>7</b>
4.1 FAShapes Namespace Reference	7
4.1.1 Detailed Description	8
4.1.2 Function Documentation	8
4.1.2.1 CreateBox()	8
4.1.2.2 CreateCone()	8
4.1.2.3 CreateCylinder()	8
4.1.2.4 CreatePyramid()	9
4.1.2.5 CreateSphere()	9
4.1.2.6 Quad()	9
<b>5 Class Documentation</b>	<b>11</b>
5.1 FAShapes::Box Class Reference	11
5.1.1 Detailed Description	12
5.1.2 Constructor & Destructor Documentation	12
5.1.2.1 Box()	12
5.1.3 Member Function Documentation	12
5.1.3.1 GetDepth()	12
5.1.3.2 GetHeight()	12
5.1.3.3 GetShape() [1/2]	12
5.1.3.4 GetShape() [2/2]	13
5.1.3.5 GetWidth()	13
5.1.3.6 InitializeBox()	13
5.1.3.7 SetDepth()	13
5.1.3.8 SetHeight()	14
5.1.3.9 SetWidth()	14
5.1.3.10 UpdateModelMatrix()	14
5.1.3.11 Volume()	14
5.2 FAShapes::Cone Class Reference	14
5.2.1 Detailed Description	15
5.2.2 Constructor & Destructor Documentation	15
5.2.2.1 Cone()	15
5.2.3 Member Function Documentation	15
5.2.3.1 GetHeight()	15
5.2.3.2 GetRadius()	16

5.2.3.3 GetShape() [1/2]	16
5.2.3.4 GetShape() [2/2]	16
5.2.3.5 InitializeCone()	16
5.2.3.6 SetHeight()	16
5.2.3.7 SetRadius()	17
5.2.3.8 UpdateModelMatrix()	17
5.2.3.9 Volume()	17
5.3 FAShapes::Cylinder Class Reference	17
5.3.1 Detailed Description	18
5.3.2 Constructor & Destructor Documentation	18
5.3.2.1 Cylinder()	18
5.3.3 Member Function Documentation	18
5.3.3.1 GetHeight()	18
5.3.3.2 GetRadius()	19
5.3.3.3 GetShape() [1/2]	19
5.3.3.4 GetShape() [2/2]	19
5.3.3.5 InitializeCylinder()	19
5.3.3.6 SetHeight()	19
5.3.3.7 SetRadius()	20
5.3.3.8 UpdateModelMatrix()	20
5.3.3.9 Volume()	20
5.4 FAShapes::Pyramid Class Reference	20
5.4.1 Detailed Description	21
5.4.2 Constructor & Destructor Documentation	21
5.4.2.1 Pyramid()	21
5.4.3 Member Function Documentation	21
5.4.3.1 GetDepth()	22
5.4.3.2 GetHeight()	22
5.4.3.3 GetShape() [1/2]	22
5.4.3.4 GetShape() [2/2]	22
5.4.3.5 GetWidth()	22
5.4.3.6 InitializePyramid()	22
5.4.3.7 SetDepth()	23
5.4.3.8 SetHeight()	23
5.4.3.9 SetWidth()	23
5.4.3.10 UpdateModelMatrix()	23
5.4.3.11 Volume()	24
5.5 FAShapes::Sphere Class Reference	24
5.5.1 Detailed Description	24
5.5.2 Constructor & Destructor Documentation	24
5.5.2.1 Sphere()	25
5.5.3 Member Function Documentation	25

5.5.3.1 GetRadius()	25
5.5.3.2 GetShape() [1/2]	25
5.5.3.3 GetShape() [2/2]	25
5.5.3.4 InitializeSphere()	25
5.5.3.5 SetRadius()	26
5.5.3.6 UpdateModelMatrix()	26
5.5.3.7 Volume()	26
5.6 FAShapes::ThreeDimensionalShape Class Reference	26
5.6.1 Constructor & Destructor Documentation	27
5.6.1.1 ThreeDimensionalShape()	27
5.6.2 Member Function Documentation	27
5.6.2.1 GetColor()	27
5.6.2.2 GetDrawArguments()	27
5.6.2.3 GetModelMatrix()	28
5.6.2.4 GetOrientation()	28
5.6.2.5 GetPosition()	28
5.6.2.6 InitializeThreeDimensionalShape()	28
5.6.2.7 RenderShape()	28
5.6.2.8 SetDrawArguments()	29
5.6.2.9 SetModelMatrix()	29
5.6.2.10 SetOrientation()	29
5.6.2.11 SetPosition()	29
5.6.2.12 UpdateShape()	29
5.7 FAShapes::Triangle Class Reference	30
5.7.1 Detailed Description	30
5.7.2 Constructor & Destructor Documentation	30
5.7.2.1 Triangle()	31
5.7.3 Member Function Documentation	31
5.7.3.1 GetCenter()	31
5.7.3.2 GetNormal()	31
5.7.3.3 GetP0()	31
5.7.3.4 GetP0Index()	32
5.7.3.5 GetP1()	32
5.7.3.6 GetP1Index()	32
5.7.3.7 GetP2()	32
5.7.3.8 GetP2Index()	32
5.7.3.9 SetP0Index()	32
5.7.3.10 SetP1Index()	33
5.7.3.11 SetP2Index()	33
5.7.3.12 SetTriangle()	33
5.7.3.13 SetTriangleIndices()	33
5.7.3.14 SetVertexList()	34

5.8 FAShapes::Vertex Struct Reference . . . . .	34
5.8.1 Detailed Description . . . . .	34
<b>6 File Documentation</b>	<b>35</b>
6.1 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FABox.h File Reference . . . . .	35
6.1.1 Detailed Description . . . . .	35
6.2 FABox.h . . . . .	36
6.3 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FACone.h File Reference . . . . .	36
6.3.1 Detailed Description . . . . .	37
6.4 FACone.h . . . . .	37
6.5 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FACylinder.h File Reference . . . . .	37
6.5.1 Detailed Description . . . . .	38
6.6 FACylinder.h . . . . .	38
6.7 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FAPyramid.h File Reference . . . . .	38
6.7.1 Detailed Description . . . . .	39
6.8 FAPyramid.h . . . . .	39
6.9 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FAShapes↵ Utility.h File Reference . . . . .	39
6.9.1 Detailed Description . . . . .	40
6.10 FAShapesUtility.h . . . . .	40
6.11 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FASphere.h File Reference . . . . .	40
6.11.1 Detailed Description . . . . .	41
6.12 FASphere.h . . . . .	41
6.13 FAThreeDimensionalShape.h . . . . .	41
6.14 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/↵ FATriangle.h File Reference . . . . .	42
6.14.1 Detailed Description . . . . .	43
6.15 FATriangle.h . . . . .	43
6.16 FAVertexStructure.h . . . . .	44
<b>Index</b>	<b>45</b>

# Chapter 1

## Namespace Index

### 1.1 Namespace List

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

<a href="#">FAShapes</a>	Has classes that are used for creating 3D shapes . . . . .	<a href="#">7</a>
--------------------------	--	-------------------





## Chapter 2

# Class Index

### 2.1 Class List

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

<a href="#">FAShapes::Box</a>	
This is class is used to create a box . . . . .	11
<a href="#">FAShapes::Cone</a>	
This is class is used to create a cone . . . . .	14
<a href="#">FAShapes::Cylinder</a>	
This is class is used to create a cylinder . . . . .	17
<a href="#">FAShapes::Pyramid</a>	
This is class is used to create a pyramid . . . . .	20
<a href="#">FAShapes::Sphere</a>	
This is class is used to create a sphere . . . . .	24
<a href="#">FAShapes::ThreeDimensionalShape</a>	26
<a href="#">FAShapes::Triangle</a>	
The class stores a pointer to a vertex list and indices to the vertices of the triangle . . . . .	30
<a href="#">FAShapes::Vertex</a>	
Data that describes a vertex . . . . .	34



## Chapter 3

# File Index

### 3.1 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/ <a href="#">FABox.h</a>	
File has a Box class under the namespace <a href="#">FAShapes</a>	35
C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/ <a href="#">FACone.h</a>	
File has a Cone class under the namespace <a href="#">FAShapes</a>	36
C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/ <a href="#">FACylinder.h</a>	
File has a Cylinder class under the namespace <a href="#">FAShapes</a>	37
C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/ <a href="#">FAPyramid.h</a>	
File has a Pyramid class under the namespace <a href="#">FAShapes</a>	38
C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/ <a href="#">FAShapesUtility.h</a>	
File has utility functions	39
C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/ <a href="#">FASphere.h</a>	
File has a Sphere class under the namespace <a href="#">FAShapes</a>	40
C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/ <a href="#">FAThreeDimensionalShape.h</a>	
<a href="#">41</a>	
C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/ <a href="#">FATriangle.h</a>	
File has a Triangle class under the namespace <a href="#">FAShapes</a>	42
C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/ <a href="#">FAVertexStructure.h</a>	
<a href="#">44</a>	



## 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 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 box's model matrix.*
- float [Volume](#) ()  
*Returns the volume of the box.*

### 5.1.1 Detailed Description

This 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	<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>position</i>	The position of the box.
in	<i>orientation</i>	The orientation of the box.
in	<i>color</i>	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	<i>width</i>	The radius of the cone.
in	<i>height</i>	The height of the cone.
in	<i>position</i>	The position of the cone.
in	<i>orientation</i>	The orientation of the cone.
in	<i>color</i>	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 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	<i>width</i>	The radius of the cylinder.
in	<i>height</i>	The height of the cylinder.
in	<i>position</i>	The position of the cylinder.
in	<i>orientation</i>	The orientation of the cylinder.
in	<i>color</i>	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 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	<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>position</i>	The position of the pyramid.
in	<i>orientation</i>	The orientation of the pyramid.
in	<i>color</i>	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 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 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	<i>radius</i>	The radius of the sphere.
in	<i>position</i>	The position of the sphere.
in	<i>orientation</i>	The orientation of the sphere.
in	<i>color</i>	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 FAShapes::Color & FAShapes::ThreeDimensionalShape::GetColor ( ) const
```

Returns the color of the 3D shape.

### 5.6.2.2 GetDrawArguments()

```
const FARender::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/FAThreeDimensionalShape.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	<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.

## 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	<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.

**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	<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.

### 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/[FAVertexStructure.h](#)



## Chapter 6

# File Documentation

### 6.1 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FAShapes/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 FBox.h

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

```

1 #pragma once
2
3 #include "FAThreeDimensionalShape.h"
4
5 namespace FAShapes
6 {
7     class Box
8     {
9     public:
10
11         Box();
12
13         void InitializeBox(float width, float height, float depth, const FAMath::Vector4D position, const
FAMath::Quaternion orientation,
14             const FAColor::Color& color);
15
16         const ThreeDimensionalShape& GetShape() const;
17
18         ThreeDimensionalShape& GetShape();
19
20         float GetWidth() const;
21
22         float GetHeight() const;
23
24         float GetDepth() const;
25
26         void SetWidth(float width);
27
28         void SetHeight(float height);
29
30         void SetDepth(float depth);
31
32         void UpdateModelMatrix();
33
34         float Volume();
35
36     private:
37         FAShapes::ThreeDimensionalShape mShape;
38
39         float mWidth;
40         float mHeight;
41         float mDepth;
42     };
43 }
```

## 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
3  #include "FAThreeDimensionalShape.h"
4
5  namespace FAShapes
6  {
7      class Cone
8      {
9      public:
10
11          Cone();
12
13          void InitializeCone(float radius, float height, const FAMath::Vector4D position, const
FAMath::Quaternion orientation,
14                          const FAColor::Color& color);
15
16          const ThreeDimensionalShape& GetShape() const;
17
18          ThreeDimensionalShape& GetShape();
19
20          float GetRadius() const;
21
22          float GetHeight() const;
23
24          void SetRadius(float radius);
25
26          void SetHeight(float height);
27
28          void UpdateModelMatrix();
29
30          float Volume();
31
32      private:
33          FAShapes::ThreeDimensionalShape mShape;
34
35          float mRadius;
36          float mHeight;
37      };
38 }
```

## 6.5 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 "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
3  #include "FAThreeDimensionalShape.h"
4
5  namespace FAShapes
6  {
7      class Cylinder
8      {
9      public:
10
11          Cylinder();
12
13          void InitializeCylinder(float radius, float height, const FAMath::Vector4D position, const
FAMath::Quaternion orientation,
14              const FAColor::Color& color);
15
16          const ThreeDimensionalShape& GetShape() const;
17
18          ThreeDimensionalShape& GetShape();
19
20          float GetRadius() const;
21
22          float GetHeight() const;
23
24          void SetRadius(float radius);
25
26          void SetHeight(float height);
27
28          void UpdateModelMatrix();
29
30          float Volume();
31
32      private:
33          FAShapes::ThreeDimensionalShape mShape;
34
35          float mRadius;
36          float mHeight;
37      };
38 }
```

## 6.7 C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FAShapes/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,
const FAColor::Color& color);
33
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>&
36         triangles,
37         unsigned int numVerticesPerCircle = 20, unsigned int numCircles = 20);
43     void CreateSphere(std::vector<FAShapes::Vertex>& vertices, std::vector<FAShapes::Triangle>&
44         triangles,
45         unsigned int numVerticesPerCircle = 20, unsigned int numCircles = 20);
50     void CreatePyramid(std::vector<FAShapes::Vertex>& vertices, std::vector<FAShapes::Triangle>&
51         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
3
4 #include "FAThreeDimensionalShape.h"
5
6 namespace FAShapes
7 {
8     class Sphere
9     {
10     public:
11
12         Sphere();
13
14         void InitializeSphere(float radius, const FAMath::Vector4D position, const FAMath::Quaternion
orientation,
15                             const FAColor::Color& color);
16
17         const ThreeDimensionalShape& GetShape() const;
18
19         ThreeDimensionalShape& GetShape();
20
21         float GetRadius() const;
22
23         void SetRadius(float radius);
24
25         void UpdateModelMatrix();
26
27         float Volume();
28
29     private:
30         FAShapes::ThreeDimensionalShape mShape;
31
32         float mRadius;
33     };
34 }
```

## 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 {
12     class Triangle
13     {
14     public:
15
16         Triangle(FAShapes::Vertex* vertexList = nullptr, unsigned int p0Index = 0, unsigned int p1Index =
17         0, unsigned int p2Index = 0);
18
19         const FAShapes::Vertex& GetP0() const;
20
21         const FAShapes::Vertex& GetP1() const;
22
23         const FAShapes::Vertex& GetP2() const;
24
25         unsigned int GetP0Index() const;
26
27         unsigned int GetP1Index() const;
28
29         unsigned int GetP2Index() const;
30
31         FAMath::Vector4D GetNormal() const;
32
33         FAMath::Vector4D GetCenter() const;
34
35         void SetVertexList(FAShapes::Vertex* vertexList);
36
37         void SetP0Index(unsigned int index);
38
39         void SetP1Index(unsigned int index);
40
41         void SetP2Index(unsigned int index);
42
43         void SetTriangleIndices(unsigned int p0Index, unsigned int p1Index, unsigned int p2Index);
44
45         void SetTriangle(FAShapes::Vertex* vertexList, unsigned int p0Index, unsigned int p1Index,
46         unsigned int p2Index);
47
48     private:
49         FAShapes::Vertex* mVertexList; //pointer to a vertex list
50         unsigned int mIndexList[3]; //indices into a vertex list
51     };
52
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 {
7     struct Vertex
8     {
9         FAMath::Vector4D position;
10        FAMath::Vector4D normal;
11        FAMath::Vector2D texCoords;
12    };
13 }
```

# Index

## Box

FAShapes::Box, 12

C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FABox.h, 35, 36

C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FACone.h, 36, 37

C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FACylinder.h, 37, 38

C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FAPyramid.h, 38, 39

C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FAShapesUtility.h, 39, 40

C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FASphere.h, 40, 41

C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FAThreeDimensionalShape.h, 41

C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FATriangle.h, 42, 43

C:/Users/Work/Desktop/First Game Engine/First-Game-Engine/FA Shapes/Header Files/FAVertexStructure.h, 44

## Cone

FAShapes::Cone, 15

## CreateBox

FAShapes, 8

## CreateCone

FAShapes, 8

## CreateCylinder

FAShapes, 8

## CreatePyramid

FAShapes, 8

## CreateSphere

FAShapes, 9

## Cylinder

FAShapes::Cylinder, 18

## FAShapes, 7

CreateBox, 8

CreateCone, 8

CreateCylinder, 8

CreatePyramid, 8

CreateSphere, 9

Quad, 9

## FAShapes::Box, 11

Box, 12

GetDepth, 12

GetHeight, 12

GetShape, 12

GetWidth, 13

InitializeBox, 13

SetDepth, 13

SetHeight, 13

SetWidth, 14

UpdateModelMatrix, 14

Volume, 14

## FAShapes::Cone, 14

Cone, 15

GetHeight, 15

GetRadius, 15

GetShape, 16

InitializeCone, 16

SetHeight, 16

SetRadius, 17

UpdateModelMatrix, 17

Volume, 17

## FAShapes::Cylinder, 17

Cylinder, 18

GetHeight, 18

GetRadius, 18

GetShape, 19

InitializeCylinder, 19

SetHeight, 19

SetRadius, 20

UpdateModelMatrix, 20

Volume, 20

## FAShapes::Pyramid, 20

GetDepth, 21

GetHeight, 22

GetShape, 22

GetWidth, 22

InitializePyramid, 22

Pyramid, 21

SetDepth, 23

SetHeight, 23

SetWidth, 23

UpdateModelMatrix, 23

Volume, 23

## FAShapes::Sphere, 24

GetRadius, 25

GetShape, 25

- InitializeSphere, [25](#)
- SetRadius, [26](#)
- Sphere, [24](#)
- UpdateModelMatrix, [26](#)
- Volume, [26](#)
- FAShapes::ThreeDimensionalShape, [26](#)
  - GetColor, [27](#)
  - GetDrawArguments, [27](#)
  - GetModelMatrix, [27](#)
  - GetOrientation, [28](#)
  - GetPosition, [28](#)
  - InitializeThreeDimensionalShape, [28](#)
  - RenderShape, [28](#)
  - SetDrawArguments, [28](#)
  - SetModelMatrix, [29](#)
  - SetOrientation, [29](#)
  - SetPosition, [29](#)
  - ThreeDimensionalShape, [27](#)
  - UpdateShape, [29](#)
- FAShapes::Triangle, [30](#)
  - GetCenter, [31](#)
  - GetNormal, [31](#)
  - GetP0, [31](#)
  - GetP0Index, [31](#)
  - GetP1, [32](#)
  - GetP1Index, [32](#)
  - GetP2, [32](#)
  - GetP2Index, [32](#)
  - SetP0Index, [32](#)
  - SetP1Index, [32](#)
  - SetP2Index, [33](#)
  - SetTriangle, [33](#)
  - SetTriangleIndices, [33](#)
  - SetVertexList, [34](#)
  - Triangle, [30](#)
- FAShapes::Vertex, [34](#)
- GetCenter
  - FAShapes::Triangle, [31](#)
- GetColor
  - FAShapes::ThreeDimensionalShape, [27](#)
- GetDepth
  - FAShapes::Box, [12](#)
  - FAShapes::Pyramid, [21](#)
- GetDrawArguments
  - FAShapes::ThreeDimensionalShape, [27](#)
- GetHeight
  - FAShapes::Box, [12](#)
  - FAShapes::Cone, [15](#)
  - FAShapes::Cylinder, [18](#)
  - FAShapes::Pyramid, [22](#)
- GetModelMatrix
  - FAShapes::ThreeDimensionalShape, [27](#)
- GetNormal
  - FAShapes::Triangle, [31](#)
- GetOrientation
  - FAShapes::ThreeDimensionalShape, [28](#)
- GetP0
  - FAShapes::Triangle, [31](#)
- GetP0Index
  - FAShapes::Triangle, [31](#)
- GetP1
  - FAShapes::Triangle, [32](#)
- GetP1Index
  - FAShapes::Triangle, [32](#)
- GetP2
  - FAShapes::Triangle, [32](#)
- GetP2Index
  - FAShapes::Triangle, [32](#)
- GetPosition
  - FAShapes::ThreeDimensionalShape, [28](#)
- GetRadius
  - FAShapes::Cone, [15](#)
  - FAShapes::Cylinder, [18](#)
  - FAShapes::Sphere, [25](#)
- GetShape
  - FAShapes::Box, [12](#)
  - FAShapes::Cone, [16](#)
  - FAShapes::Cylinder, [19](#)
  - FAShapes::Pyramid, [22](#)
  - FAShapes::Sphere, [25](#)
- GetWidth
  - FAShapes::Box, [13](#)
  - FAShapes::Pyramid, [22](#)
- InitializeBox
  - FAShapes::Box, [13](#)
- InitializeCone
  - FAShapes::Cone, [16](#)
- InitializeCylinder
  - FAShapes::Cylinder, [19](#)
- InitializePyramid
  - FAShapes::Pyramid, [22](#)
- InitializeSphere
  - FAShapes::Sphere, [25](#)
- InitializeThreeDimensionalShape
  - FAShapes::ThreeDimensionalShape, [28](#)
- Pyramid
  - FAShapes::Pyramid, [21](#)
- Quad
  - FAShapes, [9](#)
- RenderShape
  - FAShapes::ThreeDimensionalShape, [28](#)
- SetDepth
  - FAShapes::Box, [13](#)
  - FAShapes::Pyramid, [23](#)
- SetDrawArguments
  - FAShapes::ThreeDimensionalShape, [28](#)
- SetHeight
  - FAShapes::Box, [13](#)
  - FAShapes::Cone, [16](#)
  - FAShapes::Cylinder, [19](#)
  - FAShapes::Pyramid, [23](#)
- SetModelMatrix

- FAShapes::ThreeDimensionalShape, [29](#)
- SetOrientation
  - FAShapes::ThreeDimensionalShape, [29](#)
- SetP0Index
  - FAShapes::Triangle, [32](#)
- SetP1Index
  - FAShapes::Triangle, [32](#)
- SetP2Index
  - FAShapes::Triangle, [33](#)
- SetPosition
  - FAShapes::ThreeDimensionalShape, [29](#)
- SetRadius
  - FAShapes::Cone, [17](#)
  - FAShapes::Cylinder, [20](#)
  - FAShapes::Sphere, [26](#)
- SetTriangle
  - FAShapes::Triangle, [33](#)
- SetTriangleIndices
  - FAShapes::Triangle, [33](#)
- SetVertexList
  - FAShapes::Triangle, [34](#)
- SetWidth
  - FAShapes::Box, [14](#)
  - FAShapes::Pyramid, [23](#)
- Sphere
  - FAShapes::Sphere, [24](#)
- ThreeDimensionalShape
  - FAShapes::ThreeDimensionalShape, [27](#)
- Triangle
  - FAShapes::Triangle, [30](#)
- UpdateModelMatrix
  - FAShapes::Box, [14](#)
  - FAShapes::Cone, [17](#)
  - FAShapes::Cylinder, [20](#)
  - FAShapes::Pyramid, [23](#)
  - FAShapes::Sphere, [26](#)
- UpdateShape
  - FAShapes::ThreeDimensionalShape, [29](#)
- Volume
  - FAShapes::Box, [14](#)
  - FAShapes::Cone, [17](#)
  - FAShapes::Cylinder, [20](#)
  - FAShapes::Pyramid, [23](#)
  - FAShapes::Sphere, [26](#)