

Raytracer

Generated by Doxygen 1.8.15

1 Hierarchical Index	1
1.1 Class Hierarchy	1
2 Class Index	3
2.1 Class List	3
3 File Index	5
3.1 File List	5
4 Class Documentation	7
4.1 CCamera Class Reference	7
4.1.1 Detailed Description	7
4.1.2 Constructor & Destructor Documentation	8
4.1.2.1 CCamera()	8
4.1.3 Member Function Documentation	8
4.1.3.1 GetForward()	8
4.1.3.2 GetPosition()	8
4.1.3.3 GetUp()	8
4.1.3.4 RayFromUV()	9
4.1.3.5 ReadFromString()	9
4.1.3.6 SetForward()	9
4.1.3.7 SetPosition()	9
4.1.3.8 SetUp()	10
4.1.3.9 WriteToString()	10
4.2 CHeteroStore< T > Class Template Reference	10
4.2.1 Detailed Description	10
4.2.2 Constructor & Destructor Documentation	11
4.2.2.1 CHeteroStore()	11
4.2.2.2 ~CHeteroStore()	11
4.2.3 Member Function Documentation	11
4.2.3.1 operator[]() [1/2]	11
4.2.3.2 operator[]() [2/2]	11
4.2.3.3 PushBack() [1/2]	12
4.2.3.4 PushBack() [2/2]	12
4.2.3.5 Resize()	12
4.2.3.6 Size()	12
4.3 CImage Class Reference	13
4.3.1 Detailed Description	13
4.3.2 Constructor & Destructor Documentation	13
4.3.2.1 CImage() [1/2]	13
4.3.2.2 CImage() [2/2]	14
4.3.2.3 ~CImage()	14
4.3.3 Member Function Documentation	14

4.3.3.1 Blit()	14
4.3.3.2 Height()	15
4.3.3.3 operator() [1/2]	15
4.3.3.4 operator() [2/2]	15
4.3.3.5 operator=()	15
4.3.3.6 Pixels() [1/2]	15
4.3.3.7 Pixels() [2/2]	16
4.3.3.8 Read()	16
4.3.3.9 Resize()	16
4.3.3.10 Sample()	16
4.3.3.11 Width()	17
4.3.3.12 Write()	17
4.4 CMaterialDielectric Class Reference	17
4.4.1 Detailed Description	18
4.4.2 Constructor & Destructor Documentation	18
4.4.2.1 CMaterialDielectric()	18
4.4.3 Member Function Documentation	18
4.4.3.1 ReadFromString()	18
4.4.3.2 Scatter()	19
4.4.3.3 WriteToString()	19
4.5 CMaterialDiffuse Class Reference	19
4.5.1 Detailed Description	20
4.5.2 Constructor & Destructor Documentation	20
4.5.2.1 CMaterialDiffuse()	20
4.5.3 Member Function Documentation	20
4.5.3.1 ReadFromString()	20
4.5.3.2 Scatter()	21
4.5.3.3 WriteToString()	21
4.6 CMaterialMetal Class Reference	21
4.6.1 Detailed Description	22
4.6.2 Constructor & Destructor Documentation	22
4.6.2.1 CMaterialMetal()	22
4.6.3 Member Function Documentation	22
4.6.3.1 ReadFromString()	22
4.6.3.2 Scatter()	23
4.6.3.3 WriteToString()	23
4.6.4 Member Data Documentation	23
4.6.4.1 m_Fuzziness	23
4.7 CPlane Class Reference	24
4.7.1 Detailed Description	24
4.7.2 Constructor & Destructor Documentation	24
4.7.2.1 CPlane()	24

4.7.2.2 ~CPlane()	25
4.7.3 Member Function Documentation	25
4.7.3.1 GetUV()	25
4.7.3.2 Intersect()	25
4.7.3.3 ReadFromString()	26
4.7.3.4 WriteToString()	26
4.8 CRay Class Reference	26
4.8.1 Detailed Description	27
4.8.2 Constructor & Destructor Documentation	27
4.8.2.1 CRay()	27
4.8.3 Member Function Documentation	27
4.8.3.1 Direction() [1/2]	27
4.8.3.2 Direction() [2/2]	28
4.8.3.3 Origin() [1/2]	28
4.8.3.4 Origin() [2/2]	28
4.8.3.5 PointAt()	28
4.9 CScene Class Reference	28
4.9.1 Detailed Description	29
4.9.2 Constructor & Destructor Documentation	29
4.9.2.1 CScene()	29
4.9.2.2 ~CScene()	29
4.9.3 Member Function Documentation	29
4.9.3.1 AddShape()	30
4.9.3.2 Read()	30
4.9.3.3 Render()	30
4.9.3.4 RenderRegion()	30
4.9.3.5 SetCamera()	31
4.9.3.6 Write()	31
4.10 CSharedPointer< T > Class Template Reference	31
4.10.1 Detailed Description	32
4.10.2 Constructor & Destructor Documentation	32
4.10.2.1 CSharedPointer() [1/3]	32
4.10.2.2 CSharedPointer() [2/3]	32
4.10.2.3 CSharedPointer() [3/3]	32
4.10.2.4 ~CSharedPointer()	32
4.10.3 Member Function Documentation	33
4.10.3.1 IsNull()	33
4.10.3.2 operator *() [1/2]	33
4.10.3.3 operator *() [2/2]	33
4.10.3.4 operator const T *()	33
4.10.3.5 operator T *()	33
4.10.3.6 operator->() [1/2]	34

4.10.3.7 operator->() [2/2]	34
4.10.3.8 operator=()	34
4.10.3.9 Pointer() [1/2]	34
4.10.3.10 Pointer() [2/2]	34
4.11 CSphere Class Reference	35
4.11.1 Detailed Description	35
4.11.2 Constructor & Destructor Documentation	35
4.11.2.1 CSphere()	35
4.11.2.2 ~CSphere()	36
4.11.3 Member Function Documentation	36
4.11.3.1 GetUV()	36
4.11.3.2 Intersect()	36
4.11.3.3 ReadFromString()	37
4.11.3.4 WriteToString()	37
4.12 CTriangle Class Reference	37
4.12.1 Detailed Description	38
4.12.2 Constructor & Destructor Documentation	38
4.12.2.1 CTriangle()	38
4.12.2.2 ~CTriangle()	38
4.12.3 Member Function Documentation	38
4.12.3.1 GetUV()	39
4.12.3.2 Intersect()	39
4.12.3.3 ReadFromString()	39
4.12.3.4 WriteToString()	40
4.13 IMaterial Class Reference	40
4.13.1 Detailed Description	40
4.13.2 Constructor & Destructor Documentation	41
4.13.2.1 IMaterial()	41
4.13.3 Member Function Documentation	42
4.13.3.1 GetColor()	42
4.13.3.2 GetTexture()	42
4.13.3.3 Scatter()	42
4.13.4 Member Data Documentation	43
4.13.4.1 m_Color	43
4.13.4.2 m_Texture	43
4.14 ISerializable Class Reference	43
4.14.1 Detailed Description	43
4.14.2 Constructor & Destructor Documentation	44
4.14.2.1 ~ISerializable()	44
4.14.3 Member Function Documentation	44
4.14.3.1 Read()	44
4.14.3.2 Write()	44

4.15 IShape Class Reference	44
4.15.1 Detailed Description	45
4.15.2 Constructor & Destructor Documentation	45
4.15.2.1 IShape()	45
4.15.2.2 ~IShape()	45
4.15.3 Member Function Documentation	45
4.15.3.1 GetMaterial()	46
4.15.3.2 GetUV()	46
4.15.3.3 Intersect()	46
4.15.4 Member Data Documentation	46
4.15.4.1 m_Material	47
4.16 IStringSerializable Class Reference	47
4.16.1 Detailed Description	47
4.16.2 Constructor & Destructor Documentation	47
4.16.2.1 ~IStringSerializable()	47
4.16.3 Member Function Documentation	48
4.16.3.1 ReadFromString()	48
4.16.3.2 WriteToString()	48
4.17 SArguments Struct Reference	48
4.17.1 Detailed Description	48
4.17.2 Member Data Documentation	49
4.17.2.1 MaxDepth	49
4.17.2.2 MaxThreadCount	49
4.17.2.3 OutputName	49
4.17.2.4 RenderHeight	49
4.17.2.5 RenderWidth	49
4.17.2.6 SampleCount	50
4.17.2.7 ScenePath	50
4.18 SBitmapFileHeader Struct Reference	50
4.18.1 Detailed Description	50
4.18.2 Member Data Documentation	50
4.18.2.1 Offset	51
4.18.2.2 Reserved1	51
4.18.2.3 Reserved2	51
4.18.2.4 Size	51
4.18.2.5 Type	51
4.19 SBitmapInfoHeader Struct Reference	51
4.19.1 Detailed Description	52
4.19.2 Member Data Documentation	52
4.19.2.1 BitCount	52
4.19.2.2 ClrImportant	52
4.19.2.3 ClrUsed	52

4.19.2.4 Compression	53
4.19.2.5 Height	53
4.19.2.6 ImageSize	53
4.19.2.7 PixelsPerMeterX	53
4.19.2.8 PixelsPerMeterY	53
4.19.2.9 Planes	53
4.19.2.10 Size	54
4.19.2.11 Width	54
4.20 SHitInfo Struct Reference	54
4.20.1 Detailed Description	54
4.20.2 Member Data Documentation	54
4.20.2.1 Normal	54
4.20.2.2 Point	55
4.20.2.3 Shape	55
4.20.2.4 tVal	55
4.21 SRegion Struct Reference	55
4.21.1 Detailed Description	55
4.21.2 Member Data Documentation	56
4.21.2.1 Image	56
4.21.2.2 OffsetX	56
4.21.2.3 OffsetY	56
4.22 SRenderParams Struct Reference	56
4.22.1 Detailed Description	57
4.22.2 Member Data Documentation	57
4.22.2.1 AspectRatio	57
4.22.2.2 FullRenderHeight	57
4.22.2.3 FullRenderWidth	57
4.22.2.4 MaxDepth	57
4.22.2.5 MaxThreadCount	58
4.22.2.6 SampleCount	58
4.23 SSharedRenderData Struct Reference	58
4.23.1 Detailed Description	58
4.23.2 Member Data Documentation	58
4.23.2.1 PixelsProcessed	58
4.23.2.2 PrintMutex	59
4.24 UColor Union Reference	59
4.24.1 Detailed Description	59
4.24.2 Member Data Documentation	59
4.24.2.1 Alpha	59
4.24.2.2 Blue	60
4.24.2.3 Color	60
4.24.2.4 Components	60

4.24.2.5 Green	60
4.24.2.6 Red	60
4.25 Vec3 Struct Reference	61
4.25.1 Detailed Description	61
4.25.2 Constructor & Destructor Documentation	61
4.25.2.1 Vec3() [1/2]	61
4.25.2.2 Vec3() [2/2]	62
4.25.3 Member Function Documentation	62
4.25.3.1 Length()	62
4.25.3.2 LengthSq()	62
4.25.3.3 operator *() [1/2]	62
4.25.3.4 operator *() [2/2]	63
4.25.3.5 operator *=() [1/2]	63
4.25.3.6 operator *=() [2/2]	63
4.25.3.7 operator+()	63
4.25.3.8 operator+=()	63
4.25.3.9 operator-() [1/2]	64
4.25.3.10 operator-() [2/2]	64
4.25.3.11 operator-=()	64
4.25.3.12 operator/()	64
4.25.3.13 operator/=()	64
4.25.3.14 operator==()	65
4.25.4 Member Data Documentation	65
4.25.4.1 X	65
4.25.4.2 Y	65
4.25.4.3 Z	65
5 File Documentation	67
5.1 E:/dev/VS 14/Projects/raytracer/raytracer/src/camera.cpp File Reference	67
5.2 E:/dev/VS 14/Projects/raytracer/raytracer/src/camera.hpp File Reference	67
5.3 E:/dev/VS 14/Projects/raytracer/raytracer/src/color.cpp File Reference	67
5.3.1 Function Documentation	68
5.3.1.1 RGBAToU32()	68
5.3.1.2 RGBToU32()	68
5.3.1.3 U32ToVec3()	68
5.3.1.4 Vec3ToU32()	68
5.4 E:/dev/VS 14/Projects/raytracer/raytracer/src/color.hpp File Reference	69
5.4.1 Function Documentation	69
5.4.1.1 RGBAToU32()	69
5.4.1.2 RGBToU32()	69
5.4.1.3 U32ToVec3()	70
5.4.1.4 Vec3ToU32()	70

5.5 E:/dev/VS 14/Projects/raytracer/raytracer/src/common.cpp File Reference	70
5.5.1 Function Documentation	70
5.5.1.1 Cross()	71
5.5.1.2 DegreeToRadian()	71
5.5.1.3 Dot()	71
5.5.1.4 ExtractBraceContents()	71
5.5.1.5 ExtractQuote()	72
5.5.1.6 ExtractVec3()	72
5.5.1.7 Lerp()	72
5.5.1.8 Normalize()	72
5.5.1.9 operator *()	73
5.5.1.10 Project()	73
5.5.1.11 RadianToDegree()	73
5.5.1.12 RandomInUnitSphere()	73
5.5.1.13 RandomNormalized()	73
5.5.1.14 RandomNormalizedNeg()	74
5.5.1.15 Reflect()	74
5.5.1.16 Refract()	74
5.5.1.17 Reject()	74
5.5.1.18 WriteVec3()	75
5.6 E:/dev/VS 14/Projects/raytracer/raytracer/src/common.hpp File Reference	75
5.6.1 Typedef Documentation	76
5.6.1.1 r32	76
5.6.1.2 r64	76
5.6.1.3 s16	77
5.6.1.4 s32	77
5.6.1.5 s64	77
5.6.1.6 s8	77
5.6.1.7 u16	77
5.6.1.8 u32	78
5.6.1.9 u64	78
5.6.1.10 u8	78
5.6.2 Function Documentation	78
5.6.2.1 Clamp()	78
5.6.2.2 Cross()	79
5.6.2.3 DegreeToRadian()	79
5.6.2.4 Dot()	79
5.6.2.5 ExtractBraceContents()	79
5.6.2.6 ExtractQuote()	80
5.6.2.7 ExtractVec3()	80
5.6.2.8 Lerp()	80
5.6.2.9 Normalize()	80

5.6.2.10 operator *()	81
5.6.2.11 Project()	81
5.6.2.12 RadianToDegree()	81
5.6.2.13 RandomInUnitSphere()	81
5.6.2.14 RandomNormalized()	81
5.6.2.15 RandomNormalizedNeg()	82
5.6.2.16 Reflect()	82
5.6.2.17 Refract()	82
5.6.2.18 Reject()	82
5.6.2.19 WriteVec3()	83
5.6.3 Variable Documentation	83
5.6.3.1 Pi32	83
5.7 E:/dev/VS 14/Projects/raytracer/raytracer/src/heterostore.hpp File Reference	83
5.8 E:/dev/VS 14/Projects/raytracer/raytracer/src/image.cpp File Reference	83
5.9 E:/dev/VS 14/Projects/raytracer/raytracer/src/image.hpp File Reference	83
5.10 E:/dev/VS 14/Projects/raytracer/raytracer/src/material.cpp File Reference	84
5.11 E:/dev/VS 14/Projects/raytracer/raytracer/src/material.hpp File Reference	84
5.12 E:/dev/VS 14/Projects/raytracer/raytracer/src/plane.cpp File Reference	84
5.13 E:/dev/VS 14/Projects/raytracer/raytracer/src/plane.hpp File Reference	84
5.14 E:/dev/VS 14/Projects/raytracer/raytracer/src/ray.cpp File Reference	84
5.15 E:/dev/VS 14/Projects/raytracer/raytracer/src/ray.hpp File Reference	85
5.16 E:/dev/VS 14/Projects/raytracer/raytracer/src/raytracer.cpp File Reference	85
5.16.1 Function Documentation	85
5.16.1.1 main()	85
5.16.1.2 ParseArguments()	86
5.16.1.3 PrintHelp()	86
5.17 E:/dev/VS 14/Projects/raytracer/raytracer/src/scene.cpp File Reference	86
5.18 E:/dev/VS 14/Projects/raytracer/raytracer/src/scene.hpp File Reference	86
5.19 E:/dev/VS 14/Projects/raytracer/raytracer/src/serializable.cpp File Reference	87
5.19.1 Function Documentation	87
5.19.1.1 operator<<()	87
5.19.1.2 operator>>()	87
5.20 E:/dev/VS 14/Projects/raytracer/raytracer/src/serializable.hpp File Reference	87
5.20.1 Function Documentation	88
5.20.1.1 operator<<()	88
5.20.1.2 operator>>()	88
5.21 E:/dev/VS 14/Projects/raytracer/raytracer/src/shape.cpp File Reference	88
5.22 E:/dev/VS 14/Projects/raytracer/raytracer/src/shape.hpp File Reference	88
5.23 E:/dev/VS 14/Projects/raytracer/raytracer/src/sharedpointer.hpp File Reference	89
5.24 E:/dev/VS 14/Projects/raytracer/raytracer/src/sphere.cpp File Reference	89
5.25 E:/dev/VS 14/Projects/raytracer/raytracer/src/sphere.hpp File Reference	89
5.26 E:/dev/VS 14/Projects/raytracer/raytracer/src/triangle.cpp File Reference	89

5.27 E:/dev/VS 14/Projects/raytracer/raytracer/src/triangle.hpp File Reference	89
Index	91

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

CHeteroStore< T >	10
CHeteroStore< IShape >	10
CRay	26
CSharedPointer< T >	31
CSharedPointer< CImage >	31
CSharedPointer< IMaterial >	31
CSharedPointer< IShape >	31
ISerializable	43
CImage	13
CScene	28
IStringSerializable	47
CCamera	7
IMaterial	40
CMaterialDielectric	17
CMaterialDiffuse	19
CMaterialMetal	21
IShape	44
CPlane	24
CSphere	35
CTriangle	37
SArguments	48
SBitmapFileHeader	50
SBitmapInfoHeader	51
SHitInfo	54
SRegion	55
SRenderParams	56
SSharedRenderData	58
UColor	59
Vec3	61

Chapter 2

Class Index

2.1 Class List

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

CCamera	7
CHeteroStore< T >	10
CImage	13
CMaterialDielectric	17
CMaterialDiffuse	19
CMaterialMetal	21
CPlane	24
CRay	26
CScene	28
CSharedPointer< T >	31
CSphere	35
CTriangle	37
IMaterial	40
ISerializable	43
IShape	44
IStringSerializable	47
SArguments	48
SBitmapFileHeader	50
SBitmapInfoHeader	51
SHitInfo	54
SRegion	55
SRenderParams	56
SSharedRenderData	58
UColor	59
Vec3	61

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

E:/dev/VS 14/Projects/raytracer/raytracer/src/camera.cpp	67
E:/dev/VS 14/Projects/raytracer/raytracer/src/camera.hpp	67
E:/dev/VS 14/Projects/raytracer/raytracer/src/color.cpp	67
E:/dev/VS 14/Projects/raytracer/raytracer/src/color.hpp	69
E:/dev/VS 14/Projects/raytracer/raytracer/src/common.cpp	70
E:/dev/VS 14/Projects/raytracer/raytracer/src/common.hpp	75
E:/dev/VS 14/Projects/raytracer/raytracer/src/heterostore.hpp	83
E:/dev/VS 14/Projects/raytracer/raytracer/src/image.cpp	83
E:/dev/VS 14/Projects/raytracer/raytracer/src/image.hpp	83
E:/dev/VS 14/Projects/raytracer/raytracer/src/material.cpp	84
E:/dev/VS 14/Projects/raytracer/raytracer/src/material.hpp	84
E:/dev/VS 14/Projects/raytracer/raytracer/src/plane.cpp	84
E:/dev/VS 14/Projects/raytracer/raytracer/src/plane.hpp	84
E:/dev/VS 14/Projects/raytracer/raytracer/src/ray.cpp	84
E:/dev/VS 14/Projects/raytracer/raytracer/src/ray.hpp	85
E:/dev/VS 14/Projects/raytracer/raytracer/src/raytracer.cpp	85
E:/dev/VS 14/Projects/raytracer/raytracer/src/scene.cpp	86
E:/dev/VS 14/Projects/raytracer/raytracer/src/scene.hpp	86
E:/dev/VS 14/Projects/raytracer/raytracer/src/serializable.cpp	87
E:/dev/VS 14/Projects/raytracer/raytracer/src/serializable.hpp	87
E:/dev/VS 14/Projects/raytracer/raytracer/src/shape.cpp	88
E:/dev/VS 14/Projects/raytracer/raytracer/src/shape.hpp	88
E:/dev/VS 14/Projects/raytracer/raytracer/src/sharedpointer.hpp	89
E:/dev/VS 14/Projects/raytracer/raytracer/src/sphere.cpp	89
E:/dev/VS 14/Projects/raytracer/raytracer/src/sphere.hpp	89
E:/dev/VS 14/Projects/raytracer/raytracer/src/triangle.cpp	89
E:/dev/VS 14/Projects/raytracer/raytracer/src/triangle.hpp	89

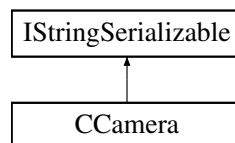
Chapter 4

Class Documentation

4.1 CCamera Class Reference

```
#include <camera.hpp>
```

Inheritance diagram for CCamera:



Public Member Functions

- [CCamera](#) ([r32](#) FieldOfView=([Pi32](#)/4.0f), [Vec3](#) Position=[Vec3](#)(0.0f, 0.0f, 0.0f), [Vec3](#) Forward=[Vec3](#)(0.0f, 0.0f, -1.0f), [Vec3](#) Up=[Vec3](#)(0.0f, 1.0f, 0.0f))
- void [SetPosition](#) ([Vec3](#) Position)
- void [SetForward](#) ([Vec3](#) Forward)
- void [SetUp](#) ([Vec3](#) Up)
- [Vec3](#) [GetPosition](#) () const
- [Vec3](#) [GetForward](#) () const
- [Vec3](#) [GetUp](#) () const
- [CRay](#) [RayFromUV](#) ([r32](#) U, [r32](#) V, [r32](#) AspectRatio=1.0f) const
- virtual std::string & [ReadFromString](#) (std::string &String) override
- virtual void [WriteToString](#) (std::string &String) const override

4.1.1 Detailed Description

Camera class

Definition at line 8 of file camera.hpp.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 CCamera()

```
CCamera::CCamera (
    r32 FieldOfView = (Pi32 / 4.0f),
    Vec3 Position = Vec3(0.0f, 0.0f, 0.0f),
    Vec3 Forward = Vec3(0.0f, 0.0f, -1.0f),
    Vec3 Up = Vec3(0.0f, 1.0f, 0.0f) )
```

Constructor

Parameters

<i>FieldOfView</i>	the camera's field of view on its vertical axis. Must be in radians.
<i>Position</i>	the camera's position in the world.
<i>Forward</i>	the direction where the camera's facing.
<i>Up</i>	the up axis of the world.

Definition at line 3 of file camera.cpp.

4.1.3 Member Function Documentation

4.1.3.1 GetForward()

```
Vec3 CCamera::GetForward ( ) const
```

Definition at line 33 of file camera.cpp.

4.1.3.2 GetPosition()

```
Vec3 CCamera::GetPosition ( ) const
```

Definition at line 29 of file camera.cpp.

4.1.3.3 GetUp()

```
Vec3 CCamera::GetUp ( ) const
```

Definition at line 37 of file camera.cpp.

4.1.3.4 RayFromUV()

```
CRay CCamera::RayFromUV (
    r32 U,
    r32 V,
    r32 AspectRatio = 1.0f ) const
```

Creates a ray from the camera to a given pixel.

Parameters

<i>U</i>	normalized horizontal pixel coordinate (0 means left, 1 means the right edge).
<i>V</i>	normalized vertical pixel coordinate (0 means bottom, 1 means upper edge).
<i>AspectRatio</i>	the image's width divided by its height.

Definition at line 42 of file camera.cpp.

4.1.3.5 ReadFromString()

```
std::string & CCamera::ReadFromString (
    std::string & String ) [override], [virtual]
```

Serializes the camera from a string (json format).

Implements [IStringSerializable](#).

Definition at line 58 of file camera.cpp.

4.1.3.6 SetForward()

```
void CCamera::SetForward (
    Vec3 Forward )
```

Definition at line 19 of file camera.cpp.

4.1.3.7 SetPosition()

```
void CCamera::SetPosition (
    Vec3 Position )
```

Definition at line 14 of file camera.cpp.

4.1.3.8 Setup()

```
void CCamera::Setup (
    Vec3 Up )
```

Definition at line 24 of file camera.cpp.

4.1.3.9 WriteToString()

```
void CCamera::WriteToString (
    std::string & String ) const [override], [virtual]
```

Serializes the camera to a string (json format).

Implements [IStringSerializable](#).

Definition at line 95 of file camera.cpp.

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

- [E:/dev/VS 14/Projects/raytracer/raytracer/src/camera.hpp](#)
- [E:/dev/VS 14/Projects/raytracer/raytracer/src/camera.cpp](#)

4.2 CHeteroStore< T > Class Template Reference

```
#include <heterostore.hpp>
```

Public Member Functions

- [CHeteroStore](#) ()
- [~CHeteroStore](#) ()
- [size_t Size](#) () const
- [void Resize](#) (size_t NewSize)
- [void PushBack](#) (T *Elem)
- [void PushBack](#) (CSharedPointer< T > Elem)
- [CSharedPointer< T > & operator\[\]](#) (size_t Index)
- [const CSharedPointer< T > & operator\[\]](#) (size_t Index) const

4.2.1 Detailed Description

```
template<class T>
class CHeteroStore< T >
```

Heterogeneous collection to store objects of the same interface but different subtypes. Non-copyable.

Definition at line 10 of file heterostore.hpp.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 CHeteroStore()

```
template<class T>
CHeteroStore< T >::CHeteroStore ( ) [inline]
```

Constructor. Creates the an empty container.

Definition at line 16 of file heterostore.hpp.

4.2.2.2 ~CHeteroStore()

```
template<class T>
CHeteroStore< T >::~~CHeteroStore ( ) [inline]
```

Definition at line 24 of file heterostore.hpp.

4.2.3 Member Function Documentation

4.2.3.1 operator[]() [1/2]

```
template<class T>
CSharedPointer<T>& CHeteroStore< T >::operator[] (
    size_t Index ) [inline]
```

Index operator to access elements. Throws, if Index is out of range.

Definition at line 77 of file heterostore.hpp.

4.2.3.2 operator[]() [2/2]

```
template<class T>
const CSharedPointer<T>& CHeteroStore< T >::operator[] (
    size_t Index ) const [inline]
```

Index operator to access elements. Throws, if Index is out of range.

Definition at line 87 of file heterostore.hpp.

4.2.3.3 PushBack() [1/2]

```
template<class T>
void CHeteroStore< T >::PushBack (
    T * Elem ) [inline]
```

Adds an element at the and of the container. Resizes the container if needed.

Definition at line 54 of file heterostore.hpp.

4.2.3.4 PushBack() [2/2]

```
template<class T>
void CHeteroStore< T >::PushBack (
    CSharedPointer< T > Elem ) [inline]
```

Adds an element at the and of the container. Resizes the container if needed.

Definition at line 66 of file heterostore.hpp.

4.2.3.5 Resize()

```
template<class T>
void CHeteroStore< T >::Resize (
    size_t NewSize ) [inline]
```

Resizes the container, copying the objects.

Definition at line 35 of file heterostore.hpp.

4.2.3.6 Size()

```
template<class T>
size_t CHeteroStore< T >::Size ( ) const [inline]
```

Definition at line 29 of file heterostore.hpp.

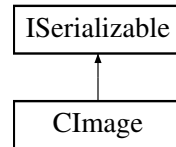
The documentation for this class was generated from the following file:

- E:/dev/VS 14/Projects/raytracer/raytracer/src/[heterostore.hpp](#)

4.3 CImage Class Reference

```
#include <image.hpp>
```

Inheritance diagram for CImage:



Public Member Functions

- [CImage](#) ([s32 Width](#)=1, [s32 Height](#)=1)
- [CImage](#) (const [CImage](#) &Other)
- virtual [~CImage](#) ()
- [CImage](#) & [operator=](#) (const [CImage](#) &Other)
- [UColor](#) & [operator\(\)](#) ([s32 X](#), [s32 Y](#))
- [UColor](#) [operator\(\)](#) ([s32 X](#), [s32 Y](#)) const
- [s32 Width](#) () const
- [s32 Height](#) () const
- [UColor](#) * [Pixels](#) ()
- const [UColor](#) * [Pixels](#) () const
- void [Resize](#) ([s32 Width](#), [s32 Height](#))
- [Vec3 Sample](#) ([r32 U](#), [r32 V](#)) const
- void [Blit](#) (const [CImage](#) &Image, [s32 OffsetX](#), [s32 OffsetY](#))
- virtual std::istream & [Read](#) (std::istream &Stream)
- virtual std::ostream & [Write](#) (std::ostream &Stream) const

4.3.1 Detailed Description

Image class that stores pixel data and allows for sampling.

Definition at line 40 of file image.hpp.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 CImage() [1/2]

```
CImage::CImage (
    s32 Width = 1,
    s32 Height = 1 )
```

Constructor

Parameters

<i>Width</i>	width of the image. Must be greater than 0.
<i>Height</i>	height of the image. Must be greate than 0.

Definition at line 5 of file image.cpp.

4.3.2.2 CImage() [2/2]

```
CImage::CImage (
    const CImage & Other )
```

Definition at line 17 of file image.cpp.

4.3.2.3 ~CImage()

```
CImage::~CImage ( ) [virtual]
```

Definition at line 27 of file image.cpp.

4.3.3 Member Function Documentation**4.3.3.1 Blit()**

```
void CImage::Blit (
    const CImage & Image,
    s32 OffsetX,
    s32 OffsetY )
```

Copies an image to another at a given location.

Parameters

<i>Image</i>	the image to copy to this image.
<i>OffsetX</i>	the X coordinate at which to start the copy.
<i>OffsetY</i>	the Y coordinate at which to start the copy.

Definition at line 113 of file image.cpp.

4.3.3.2 Height()

```
s32 CImage::Height ( ) const
```

Definition at line 74 of file image.cpp.

4.3.3.3 operator() [1/2]

```
UColor & CImage::operator() (
    s32 X,
    s32 Y )
```

Returns the pixel at (X, Y). Throws out_of_range exception if X or Y are invalid.

Definition at line 48 of file image.cpp.

4.3.3.4 operator() [2/2]

```
UColor CImage::operator() (
    s32 X,
    s32 Y ) const
```

Returns the pixel at (X, Y). Throws out_of_range exception if X or Y are invalid.

Definition at line 59 of file image.cpp.

4.3.3.5 operator=()

```
CImage & CImage::operator= (
    const CImage & Other )
```

Definition at line 32 of file image.cpp.

4.3.3.6 Pixels() [1/2]

```
UColor * CImage::Pixels ( )
```

Returns the raw pointer to the pixel data.

Definition at line 79 of file image.cpp.

4.3.3.7 Pixels() [2/2]

```
const UColor * CImage::Pixels ( ) const
```

Returns the raw pointer to the pixel data.

Definition at line 84 of file image.cpp.

4.3.3.8 Read()

```
std::istream & CImage::Read (
    std::istream & Stream ) [virtual]
```

Reads the image in .bmp format from a stream.

Parameters

<i>Stream</i>	the stream from which to read. Must be binary.
---------------	--

Implements [ISerializable](#).

Definition at line 133 of file image.cpp.

4.3.3.9 Resize()

```
void CImage::Resize (
    s32 Width,
    s32 Height )
```

Resizes the image to a new resolution

Parameters

<i>Width</i>	the new width of the image. Must be greater than 0.
<i>Height</i>	the new height of the iamge. Must be reater than 0.

Definition at line 89 of file image.cpp.

4.3.3.10 Sample()

```
Vec3 CImage::Sample (
    r32 U,
    r32 V ) const
```

Returns a normalized color from the normalized image coordinates.

Parameters

<i>U</i>	normalized horizontal coordinate ranging from [0..1).
<i>V</i>	normalized vertical coordinate ranging from [0..1).

Definition at line 103 of file image.cpp.

4.3.3.11 Width()

```
s32 CImage::Width ( ) const
```

Definition at line 70 of file image.cpp.

4.3.3.12 Write()

```
std::ostream & CImage::Write (
    std::ostream & Stream ) const [virtual]
```

Writes the image in .bmp format to a stream.

Parameters

<i>Stream</i>	the stream to write. Must be binary.
---------------	--------------------------------------

Implements [ISerializable](#).

Definition at line 176 of file image.cpp.

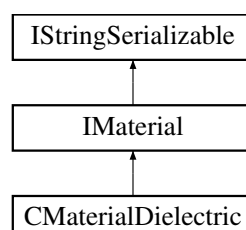
The documentation for this class was generated from the following files:

- E:/dev/VS 14/Projects/raytracer/raytracer/src/image.hpp
- E:/dev/VS 14/Projects/raytracer/raytracer/src/image.cpp

4.4 CMaterialDielectric Class Reference

```
#include <material.hpp>
```

Inheritance diagram for CMaterialDielectric:



Public Member Functions

- [CMaterialDielectric](#) ([Vec3](#) Color, [r32](#) RefractiveIndex, [CSharedPointer](#)< [CImage](#) > Texture=nullptr)
- virtual [CRay Scatter](#) (const [CRay](#) &Ray, [Vec3](#) Position, [Vec3](#) Normal) const
- virtual std::string & [ReadFromString](#) (std::string &String)
- virtual void [WriteToString](#) (std::string &String) const

Additional Inherited Members

4.4.1 Detailed Description

Dielectric material which rays pass through.

Definition at line 72 of file material.hpp.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 CMaterialDielectric()

```
CMaterialDielectric::CMaterialDielectric (
    Vec3 Color,
    r32 RefractiveIndex,
    CSharedPointer< CImage > Texture = nullptr )
```

See [IMaterial](#)

Parameters

<i>Color</i>	the color of the material.
<i>RefractiveIndex</i>	the physical refractive index of the material.
<i>Texture</i>	the texture of the material.

Definition at line 143 of file material.cpp.

4.4.3 Member Function Documentation

4.4.3.1 ReadFromString()

```
std::string & CMaterialDielectric::ReadFromString (
    std::string & String ) [virtual]
```

Reads the object from a string, possibly removing contents from the string.

Implements [IStringSerializable](#).

Definition at line 175 of file material.cpp.

4.4.3.2 Scatter()

```
CRay CMaterialDielectric::Scatter (  
    const CRay & Ray,  
    Vec3 Position,  
    Vec3 Normal ) const [virtual]
```

Returns the refraction/reflection of the ray depending on the angle.

Implements [IMaterial](#).

Definition at line 151 of file material.cpp.

4.4.3.3 WriteToString()

```
void CMaterialDielectric::WriteToString (  
    std::string & String ) const [virtual]
```

Writes the object to a string.

Implements [IStringSerializable](#).

Definition at line 213 of file material.cpp.

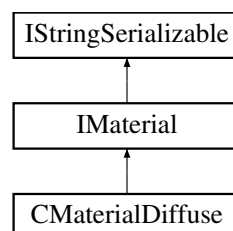
The documentation for this class was generated from the following files:

- E:/dev/VS 14/Projects/raytracer/raytracer/src/[material.hpp](#)
- E:/dev/VS 14/Projects/raytracer/raytracer/src/[material.cpp](#)

4.5 CMaterialDiffuse Class Reference

```
#include <material.hpp>
```

Inheritance diagram for CMaterialDiffuse:



Public Member Functions

- [CMaterialDiffuse](#) ([Vec3](#) Color, [CSharedPointer](#)< [CImage](#) > Texture=nullptr)
- virtual [CRay Scatter](#) (const [CRay](#) &Ray, [Vec3](#) Position, [Vec3](#) Normal) const
- virtual std::string & [ReadFromString](#) (std::string &String)
- virtual void [WriteToString](#) (std::string &String) const

Additional Inherited Members

4.5.1 Detailed Description

Diffuse (matte) material

Definition at line 38 of file material.hpp.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 CMaterialDiffuse()

```
CMaterialDiffuse::CMaterialDiffuse (
    Vec3 Color,
    CSharedPointer< CImage > Texture = nullptr )
```

See [IMaterial](#)

Definition at line 19 of file material.cpp.

4.5.3 Member Function Documentation

4.5.3.1 ReadFromString()

```
std::string & CMaterialDiffuse::ReadFromString (
    std::string & String ) [virtual]
```

Reads the object from a string, possibly removing contents from the string.

Implements [IStringSerializable](#).

Definition at line 30 of file material.cpp.

4.5.3.2 Scatter()

```
CRay CMaterialDiffuse::Scatter (
    const CRay & Ray,
    Vec3 Position,
    Vec3 Normal ) const [virtual]
```

Scatters the ray in a random direction.

Implements [IMaterial](#).

Definition at line 25 of file material.cpp.

4.5.3.3 WriteToString()

```
void CMaterialDiffuse::WriteToString (
    std::string & String ) const [virtual]
```

Writes the object to a string.

Implements [IStringSerializable](#).

Definition at line 60 of file material.cpp.

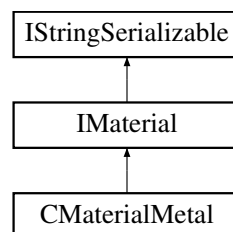
The documentation for this class was generated from the following files:

- E:/dev/VS 14/Projects/raytracer/raytracer/src/[material.hpp](#)
- E:/dev/VS 14/Projects/raytracer/raytracer/src/[material.cpp](#)

4.6 CMaterialMetal Class Reference

```
#include <material.hpp>
```

Inheritance diagram for CMaterialMetal:



Public Member Functions

- [CMaterialMetal](#) ([Vec3](#) Color, [r32](#) Fuzziness=0.0f, [CSharedPointer](#)< [CImage](#) > Texture=nullptr)
- virtual [CRay Scatter](#) (const [CRay](#) &Ray, [Vec3](#) Position, [Vec3](#) Normal) const
- virtual std::string & [ReadFrom](#)[String](#) (std::string &String)
- virtual void [Write](#)[ToString](#) (std::string &String) const

Protected Attributes

- [r32 m_Fuzziness](#)

4.6.1 Detailed Description

Metallic material which reflects rays in a mirror-life fashion.

Definition at line 52 of file material.hpp.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 CMaterialMetal()

```
CMaterialMetal::CMaterialMetal (
    Vec3 Color,
    r32 Fuzziness = 0.0f,
    CSharedPointer< CImage > Texture = nullptr )
```

See [IMaterial](#)

Parameters

<i>Color</i>	the color of the material.
<i>Fuzziness</i>	randomizes the direction of the reflection. 0 means perfectly clear metal.
<i>Texture</i>	the texture of the material.

Definition at line 75 of file material.cpp.

4.6.3 Member Function Documentation

4.6.3.1 ReadFromString()

```
std::string & CMaterialMetal::ReadFromString (
    std::string & String ) [virtual]
```

Reads the object from a string, possibly removing contents from the string.

Implements [IStringSerializable](#).

Definition at line 89 of file material.cpp.

4.6.3.2 Scatter()

```
CRay CMaterialMetal::Scatter (
    const CRay & Ray,
    Vec3 Position,
    Vec3 Normal ) const [virtual]
```

Mathematically reflects ray, randomizing it by the metal's fuzziness.

Implements [IMaterial](#).

Definition at line 82 of file material.cpp.

4.6.3.3 WriteToString()

```
void CMaterialMetal::WriteToString (
    std::string & String ) const [virtual]
```

Writes the object to a string.

Implements [IStringSerializable](#).

Definition at line 127 of file material.cpp.

4.6.4 Member Data Documentation

4.6.4.1 m_Fuzziness

```
r32 CMaterialMetal::m_Fuzziness [protected]
```

Parameter which controls how fuzzy the metal is. 0 means perfectly clear.

Definition at line 68 of file material.hpp.

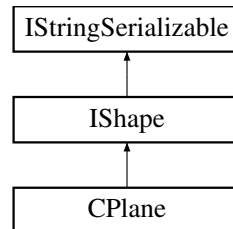
The documentation for this class was generated from the following files:

- E:/dev/VS 14/Projects/raytracer/raytracer/src/[material.hpp](#)
- E:/dev/VS 14/Projects/raytracer/raytracer/src/[material.cpp](#)

4.7 CPlane Class Reference

```
#include <plane.hpp>
```

Inheritance diagram for CPlane:



Public Member Functions

- **CPlane** (**Vec3** Normal, **r32** Offset, **Vec3** TextureUp=**Vec3**(0.0f, 0.0f, -1.0f), CSharedPointer< **IMaterial** > Material=nullptr)
- virtual **~CPlane** ()
- virtual void **GetUV** (**Vec3** Point, **r32** &U, **r32** &V) const
- virtual bool **Intersect** (const **CRay** &Ray, **r32** tMin, **r32** tMax, **SHitInfo** &HitInfo) const
- virtual std::string & **ReadFromString** (std::string &String)
- virtual void **WriteToString** (std::string &String) const

Additional Inherited Members

4.7.1 Detailed Description

Shape that represents a plane in the world.

Equation for the plane is $N_x \cdot x + N_y \cdot y + N_z \cdot z = \text{offset}$

Definition at line 10 of file plane.hpp.

4.7.2 Constructor & Destructor Documentation

4.7.2.1 CPlane()

```

CPlane::CPlane (
    Vec3 Normal,
    r32 Offset,
    Vec3 TextureUp = Vec3(0.0f, 0.0f, -1.0f),
    CSharedPointer< IMaterial > Material = nullptr )
  
```

Constructor

Parameters

<i>Normal</i>	the surface normal of the plane.
<i>Offset</i>	the value how far along the plane is on the normal relative to the world origin.
<i>TextureUp</i>	world vector which defines which axis corresponds to the texture's vertical axis.
<i>Material</i>	the material of the plane.

Definition at line 3 of file plane.cpp.

4.7.2.2 ~CPlane()

```
CPlane::~CPlane ( ) [virtual]
```

Definition at line 15 of file plane.cpp.

4.7.3 Member Function Documentation

4.7.3.1 GetUV()

```
void CPlane::GetUV (
    Vec3 Point,
    r32 & U,
    r32 & V ) const [virtual]
```

Returns the UV coordinates of the object at a given point.

Implements [IShape](#).

Definition at line 20 of file plane.cpp.

4.7.3.2 Intersect()

```
bool CPlane::Intersect (
    const CRay & Ray,
    r32 tMin,
    r32 tMax,
    SHitInfo & HitInfo ) const [virtual]
```

Checks whether a ray intersects with the shape.

Parameters

<i>Ray</i>	the ray to check the intersection with.
<i>tMin</i>	the minimum t parameter of the ray to consider for intersection.
<i>tMax</i>	the maximum t paramter of the ray to considered for intersection.
<i>HitInfo</i>	reference to the object which will store the collision information.

Implements [IShape](#).

Definition at line 26 of file plane.cpp.

4.7.3.3 ReadFromString()

```
std::string & CPlane::ReadFromString (
    std::string & String ) [virtual]
```

Reads the object from a string, possibly removing contents from the string.

Implements [IStringSerializable](#).

Definition at line 47 of file plane.cpp.

4.7.3.4 WriteToString()

```
void CPlane::WriteToString (
    std::string & String ) const [virtual]
```

Writes the object to a string.

Implements [IStringSerializable](#).

Definition at line 105 of file plane.cpp.

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

- E:/dev/VS 14/Projects/raytracer/raytracer/src/[plane.hpp](#)
- E:/dev/VS 14/Projects/raytracer/raytracer/src/[plane.cpp](#)

4.8 CRay Class Reference

```
#include <ray.hpp>
```

Public Member Functions

- [CRay](#) ([Vec3](#) *Origin*, [Vec3](#) *Direction*)
- [Vec3](#) & [Origin](#) ()
- [Vec3](#) [Origin](#) () const
- [Vec3](#) & [Direction](#) ()
- [Vec3](#) [Direction](#) () const
- [Vec3](#) [PointAt](#) ([r32](#) *tVal*) const

4.8.1 Detailed Description

Class that represents a physical ray, that has a starting point and a direction.

Definition at line 6 of file ray.hpp.

4.8.2 Constructor & Destructor Documentation

4.8.2.1 CRay()

```
CRay::CRay (
    Vec3 Origin,
    Vec3 Direction )
```

Constructor

Parameters

<i>Origin</i>	starting location of the ray in the world.
<i>Direction</i>	direction vector of the ray.

Definition at line 3 of file ray.cpp.

4.8.3 Member Function Documentation

4.8.3.1 Direction() [1/2]

```
Vec3 & CRay::Direction ( )
```

Definition at line 19 of file ray.cpp.

4.8.3.2 Direction() [2/2]

```
Vec3 CRay::Direction ( ) const
```

Definition at line 23 of file ray.cpp.

4.8.3.3 Origin() [1/2]

```
Vec3 & CRay::Origin ( )
```

Definition at line 10 of file ray.cpp.

4.8.3.4 Origin() [2/2]

```
Vec3 CRay::Origin ( ) const
```

Definition at line 14 of file ray.cpp.

4.8.3.5 PointAt()

```
Vec3 CRay::PointAt (
    r32 tVal ) const
```

Returns a position along the ray given a parameter.

The position returned is $P(t) = A + V * t$.

Parameters

<i>tVal</i>	the parameter.
-------------	----------------

Definition at line 28 of file ray.cpp.

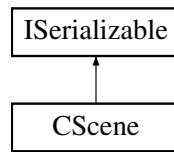
The documentation for this class was generated from the following files:

- [E:/dev/VS 14/Projects/raytracer/raytracer/src/ray.hpp](#)
- [E:/dev/VS 14/Projects/raytracer/raytracer/src/ray.cpp](#)

4.9 CScene Class Reference

```
#include <scene.hpp>
```


Inheritance diagram for CScene:



Public Member Functions

- [CScene](#) ()
- [~CScene](#) ()
- void [SetCamera](#) ([CCamera](#) Camera)
- void [AddShape](#) ([IShape](#) *Shape)
- void [RenderRegion](#) ([SRegion](#) &Region, [SRenderParams](#) &Params, [SSharedRenderData](#) &Shared) const
- void [Render](#) ([CImage](#) &Image, [SRenderParams](#) &Params) const
- virtual std::istream & [Read](#) (std::istream &Stream)
- virtual std::ostream & [Write](#) (std::ostream &Stream) const

4.9.1 Detailed Description

Scene class that contains the objects, camera and renders the image.

Definition at line 41 of file scene.hpp.

4.9.2 Constructor & Destructor Documentation

4.9.2.1 CScene()

```
CScene::CScene ( )
```

Definition at line 9 of file scene.cpp.

4.9.2.2 ~CScene()

```
CScene::~~CScene ( )
```

Definition at line 12 of file scene.cpp.

4.9.3 Member Function Documentation

4.9.3.1 AddShape()

```
void CScene::AddShape (
    IShape * Shape )
```

Adds a shape to the scene.

Definition at line 22 of file scene.cpp.

4.9.3.2 Read()

```
std::istream & CScene::Read (
    std::istream & Stream ) [virtual]
```

Reads the scene from a stream in json format.

Implements [ISerializable](#).

Definition at line 209 of file scene.cpp.

4.9.3.3 Render()

```
void CScene::Render (
    CImage & Image,
    SRenderParams & Params ) const
```

Renders the scene to an image.

Parameters

<i>Image</i>	the image to render to.
<i>Params</i>	the render parameters.

Definition at line 128 of file scene.cpp.

4.9.3.4 RenderRegion()

```
void CScene::RenderRegion (
    SRegion & Region,
    SRenderParams & Params,
    SSharedRenderData & Shared ) const
```

Renders a subregion of the final image.

Parameters

<i>Region</i>	the region to render.
<i>Params</i>	the render parameters.
<i>Shared</i>	the data shared between threads.

Definition at line 77 of file scene.cpp.

4.9.3.5 SetCamera()

```
void CScene::SetCamera (
    CCamera Camera )
```

Definition at line 17 of file scene.cpp.

4.9.3.6 Write()

```
std::ostream & CScene::Write (
    std::ostream & Stream ) const [virtual]
```

Writes the scene to a stream in json format.

Implements [ISerializable](#).

Definition at line 252 of file scene.cpp.

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

- E:/dev/VS 14/Projects/raytracer/raytracer/src/[scene.hpp](#)
- E:/dev/VS 14/Projects/raytracer/raytracer/src/[scene.cpp](#)

4.10 CSharedPointer< T > Class Template Reference

```
#include <sharedpointer.hpp>
```

Public Member Functions

- [CSharedPointer](#) ()
- [CSharedPointer](#) (T *[Pointer](#))
- [CSharedPointer](#) (const [CSharedPointer](#) &Other)
- [~CSharedPointer](#) ()
- [CSharedPointer](#) & [operator=](#) (const [CSharedPointer](#) &Other)
- [operator T *](#) ()
- [operator const T *](#) () const
- T & [operator *](#) ()
- const T & [operator *](#) () const
- T * [operator->](#) ()
- const T * [operator->](#) () const
- T * [Pointer](#) ()
- const T * [Pointer](#) () const
- bool [IsNull](#) () const

4.10.1 Detailed Description

```
template<class T>
class CSharedPointer< T >
```

Shared pointer class which uses reference counting to keep track of its objects.

Definition at line 7 of file sharedpointer.hpp.

4.10.2 Constructor & Destructor Documentation

4.10.2.1 CSharedPointer() [1/3]

```
template<class T>
CSharedPointer< T >::CSharedPointer ( ) [inline]
```

Definition at line 10 of file sharedpointer.hpp.

4.10.2.2 CSharedPointer() [2/3]

```
template<class T>
CSharedPointer< T >::CSharedPointer (
    T * Pointer ) [inline]
```

Definition at line 16 of file sharedpointer.hpp.

4.10.2.3 CSharedPointer() [3/3]

```
template<class T>
CSharedPointer< T >::CSharedPointer (
    const CSharedPointer< T > & Other ) [inline]
```

Definition at line 26 of file sharedpointer.hpp.

4.10.2.4 ~CSharedPointer()

```
template<class T>
CSharedPointer< T >::~~CSharedPointer ( ) [inline]
```

Definition at line 36 of file sharedpointer.hpp.

4.10.3 Member Function Documentation

4.10.3.1 IsNull()

```
template<class T>
bool CSharedPointer< T >::IsNull ( ) const [inline]
```

Definition at line 112 of file sharedpointer.hpp.

4.10.3.2 operator*() [1/2]

```
template<class T>
T& CSharedPointer< T >::operator * ( ) [inline]
```

Definition at line 82 of file sharedpointer.hpp.

4.10.3.3 operator*() [2/2]

```
template<class T>
const T& CSharedPointer< T >::operator * ( ) const [inline]
```

Definition at line 87 of file sharedpointer.hpp.

4.10.3.4 operator const T*()

```
template<class T>
CSharedPointer< T >::operator const T * ( ) const [inline]
```

Definition at line 77 of file sharedpointer.hpp.

4.10.3.5 operator T*()

```
template<class T>
CSharedPointer< T >::operator T * ( ) [inline]
```

Definition at line 72 of file sharedpointer.hpp.

4.10.3.6 operator->() [1/2]

```
template<class T>
T* CSharedPointer< T >::operator-> ( ) [inline]
```

Definition at line 92 of file sharedpointer.hpp.

4.10.3.7 operator->() [2/2]

```
template<class T>
const T* CSharedPointer< T >::operator-> ( ) const [inline]
```

Definition at line 97 of file sharedpointer.hpp.

4.10.3.8 operator=()

```
template<class T>
CSharedPointer& CSharedPointer< T >::operator= (
    const CSharedPointer< T > & Other ) [inline]
```

Definition at line 49 of file sharedpointer.hpp.

4.10.3.9 Pointer() [1/2]

```
template<class T>
T* CSharedPointer< T >::Pointer ( ) [inline]
```

Definition at line 102 of file sharedpointer.hpp.

4.10.3.10 Pointer() [2/2]

```
template<class T>
const T* CSharedPointer< T >::Pointer ( ) const [inline]
```

Definition at line 107 of file sharedpointer.hpp.

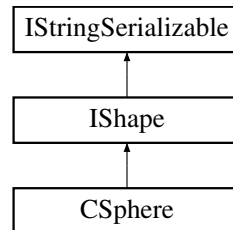
The documentation for this class was generated from the following file:

- E:/dev/VS 14/Projects/raytracer/raytracer/src/[sharedpointer.hpp](#)

4.11 CSphere Class Reference

```
#include <sphere.hpp>
```

Inheritance diagram for CSphere:



Public Member Functions

- [CSphere](#) ([Vec3](#) Center, [r32](#) Radius, [CSharedPointer](#)< [IMaterial](#) > Material=nullptr)
- virtual [~CSphere](#) ()
- virtual void [GetUV](#) ([Vec3](#) Point, [r32](#) &U, [r32](#) &V) const
- virtual bool [Intersect](#) (const [CRay](#) &Ray, [r32](#) tMin, [r32](#) tMax, [SHitInfo](#) &HitInfo) const
- virtual std::string & [ReadFromString](#) (std::string &String)
- virtual void [WriteToString](#) (std::string &String) const

Additional Inherited Members

4.11.1 Detailed Description

Shape class that represents a sphere in the world.

Definition at line 5 of file sphere.hpp.

4.11.2 Constructor & Destructor Documentation

4.11.2.1 CSphere()

```
CSphere::CSphere (
    Vec3 Center,
    r32 Radius,
    CSharedPointer< IMaterial > Material = nullptr )
```

Constructor

Parameters

<i>Center</i>	the location of the sphere.
<i>Radius</i>	the radius of the sphere.
<i>Material</i>	the material of the sphere.

Definition at line 3 of file sphere.cpp.

4.11.2.2 ~CSphere()

```
CSphere::~CSphere ( ) [virtual]
```

Definition at line 10 of file sphere.cpp.

4.11.3 Member Function Documentation

4.11.3.1 GetUV()

```
void CSphere::GetUV (
    Vec3 Point,
    r32 & U,
    r32 & V ) const [virtual]
```

Returns the UV coordinates of the object at a given point.

Implements [IShape](#).

Definition at line 15 of file sphere.cpp.

4.11.3.2 Intersect()

```
bool CSphere::Intersect (
    const CRay & Ray,
    r32 tMin,
    r32 tMax,
    SHitInfo & HitInfo ) const [virtual]
```

Checks whether a ray intersects with the shape.

Parameters

<i>Ray</i>	the ray to check the intersection with.
<i>tMin</i>	the minimum t parameter of the ray to consider for intersection.
<i>tMax</i>	the maximum t paramter of the ray to considered for intersection.
<i>HitInfo</i>	reference to the object which will store the collision information.

Implements [IShape](#).

Definition at line 22 of file sphere.cpp.

4.11.3.3 ReadFromString()

```
std::string & CSphere::ReadFromString (
    std::string & String ) [virtual]
```

Reads the object from a string, possibly removing contents from the string.

Implements [IStringSerializable](#).

Definition at line 50 of file sphere.cpp.

4.11.3.4 WriteToString()

```
void CSphere::WriteToString (
    std::string & String ) const [virtual]
```

Writes the object to a string.

Implements [IStringSerializable](#).

Definition at line 98 of file sphere.cpp.

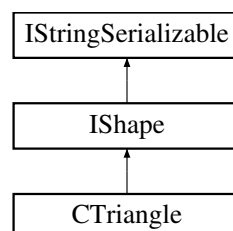
The documentation for this class was generated from the following files:

- [E:/dev/VS 14/Projects/raytracer/raytracer/src/sphere.hpp](#)
- [E:/dev/VS 14/Projects/raytracer/raytracer/src/sphere.cpp](#)

4.12 CTriangle Class Reference

```
#include <triangle.hpp>
```

Inheritance diagram for CTriangle:



Public Member Functions

- [CTriangle](#) ([CSharedPointer](#)< [IMaterial](#) > *Material*, [Vec3](#) *P0*, [Vec3](#) *P1*, [Vec3](#) *P2*, bool *bCustomNormals*=false, [Vec3](#) *N0*=[Vec3](#)(), [Vec3](#) *N1*=[Vec3](#)(), [Vec3](#) *N2*=[Vec3](#)())
- virtual [~CTriangle](#) ()
- virtual void [GetUV](#) ([Vec3](#) *Point*, [r32](#) &*U*, [r32](#) &*V*) const
- virtual bool [Intersect](#) (const [CRay](#) &*Ray*, [r32](#) *tMin*, [r32](#) *tMax*, [SHitInfo](#) &*HitInfo*) const
- virtual std::string & [ReadFromString](#) (std::string &*String*)
- virtual void [WriteToString](#) (std::string &*String*) const

Additional Inherited Members

4.12.1 Detailed Description

Definition at line 5 of file `triangle.hpp`.

4.12.2 Constructor & Destructor Documentation

4.12.2.1 CTriangle()

```
CTriangle::CTriangle (
    CSharedPointer< IMaterial > Material,
    Vec3 P0,
    Vec3 P1,
    Vec3 P2,
    bool bCustomNormals = false,
    Vec3 N0 = Vec3(),
    Vec3 N1 = Vec3(),
    Vec3 N2 = Vec3() )
```

Definition at line 4 of file `triangle.cpp`.

4.12.2.2 ~CTriangle()

```
CTriangle::~CTriangle ( ) [virtual]
```

Definition at line 30 of file `triangle.cpp`.

4.12.3 Member Function Documentation

4.12.3.1 GetUV()

```
void CTriangle::GetUV (
    Vec3 Point,
    r32 & U,
    r32 & V ) const [virtual]
```

Returns the UV coordinates of the object at a given point.

Implements [IShape](#).

Definition at line 35 of file triangle.cpp.

4.12.3.2 Intersect()

```
bool CTriangle::Intersect (
    const CRay & Ray,
    r32 tMin,
    r32 tMax,
    SHitInfo & HitInfo ) const [virtual]
```

Checks whether a ray intersects with the shape.

Parameters

<i>Ray</i>	the ray to check the intersection with.
<i>tMin</i>	the minimum t parameter of the ray to consider for intersection.
<i>tMax</i>	the maximum t paramter of the ray to considered for intersection.
<i>HitInfo</i>	reference to the object which will store the collision information.

Implements [IShape](#).

Definition at line 41 of file triangle.cpp.

4.12.3.3 ReadFromString()

```
std::string & CTriangle::ReadFromString (
    std::string & String ) [virtual]
```

Reads the object from a string, possibly removing contents from the string.

Implements [IStringSerializable](#).

Definition at line 83 of file triangle.cpp.

4.12.3.4 WriteToString()

```
void CTriangle::WriteToString (
    std::string & String ) const [virtual]
```

Writes the object to a string.

Implements [IStringSerializable](#).

Definition at line 88 of file triangle.cpp.

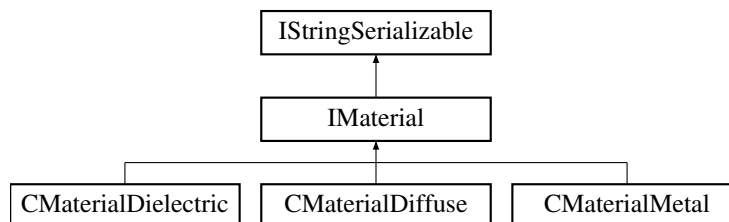
The documentation for this class was generated from the following files:

- E:/dev/VS 14/Projects/raytracer/raytracer/src/triangle.hpp
- E:/dev/VS 14/Projects/raytracer/raytracer/src/triangle.cpp

4.13 IMaterial Class Reference

```
#include <material.hpp>
```

Inheritance diagram for IMaterial:



Public Member Functions

- [IMaterial](#) ([Vec3](#) Color, [CSharedPointer](#)< [CImage](#) > Texture=nullptr)
- [Vec3](#) [GetColor](#) () const
- const [CSharedPointer](#)< [CImage](#) > & [GetTexture](#) () const
- virtual [CRay](#) [Scatter](#) (const [CRay](#) &Ray, [Vec3](#) Position, [Vec3](#) Normal) const =0

Protected Attributes

- [Vec3](#) m_Color
- [CSharedPointer](#)< [CImage](#) > m_Texture

4.13.1 Detailed Description

Generic material interface. Stores an object's color, texture and the reflection function.

Definition at line 11 of file material.hpp.

4.13.2 Constructor & Destructor Documentation

4.13.2.1 IMaterial()

```
IMaterial::IMaterial (
    Vec3 Color,
    CSharedPointer< CImage > Texture = nullptr )
```

Constructor

Parameters

<i>Color</i>	the material's diffuse color.
<i>Texture</i>	pointer to a CImage which contains additional color information.

Definition at line 3 of file material.cpp.

4.13.3 Member Function Documentation

4.13.3.1 GetColor()

```
Vec3 IMaterial::GetColor ( ) const
```

Returns the material's color.

Definition at line 9 of file material.cpp.

4.13.3.2 GetTexture()

```
const CSharedPointer< CImage > & IMaterial::GetTexture ( ) const
```

Returns a pointer to material's texture.

Definition at line 14 of file material.cpp.

4.13.3.3 Scatter()

```
virtual CRay IMaterial::Scatter (
    const CRay & Ray,
    Vec3 Position,
    Vec3 Normal ) const [pure virtual]
```

Reflects the vector given the surface coordinate and its normal.

Parameters

<i>Ray</i>	the ray to reflect.
<i>Position</i>	the coordinate where the collision occurred.
<i>Normal</i>	the surface normal of the shape where the collision occurred.

Implemented in [CMaterialDielectric](#), [CMaterialMetal](#), and [CMaterialDiffuse](#).

4.13.4 Member Data Documentation

4.13.4.1 m_Color

`Vec3 IMaterial::m_Color [protected]`

Definition at line 33 of file material.hpp.

4.13.4.2 m_Texture

`CSharedPointer<CImage> IMaterial::m_Texture [protected]`

Definition at line 34 of file material.hpp.

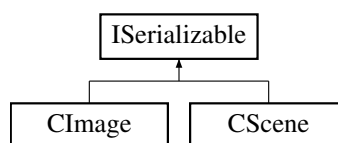
The documentation for this class was generated from the following files:

- E:/dev/VS 14/Projects/raytracer/raytracer/src/material.hpp
- E:/dev/VS 14/Projects/raytracer/raytracer/src/material.cpp

4.14 ISerializable Class Reference

```
#include <serializable.hpp>
```

Inheritance diagram for ISerializable:



Public Member Functions

- virtual `~ISerializable()`
- virtual `std::istream & Read (std::istream &Stream)=0`
- virtual `std::ostream & Write (std::ostream &Stream) const =0`

4.14.1 Detailed Description

Serializable interface that supports reading from and writing to streams.

Definition at line 6 of file serializable.hpp.

4.14.2 Constructor & Destructor Documentation

4.14.2.1 ~ISerializable()

```
ISerializable::~~ISerializable ( ) [virtual]
```

Definition at line 3 of file serializable.cpp.

4.14.3 Member Function Documentation

4.14.3.1 Read()

```
virtual std::istream& ISerializable::Read (
    std::istream & Stream ) [pure virtual]
```

Reads the object from a stream.

Implemented in [CImage](#), and [CScene](#).

4.14.3.2 Write()

```
virtual std::ostream& ISerializable::Write (
    std::ostream & Stream ) const [pure virtual]
```

Writes the object to a stream.

Implemented in [CImage](#), and [CScene](#).

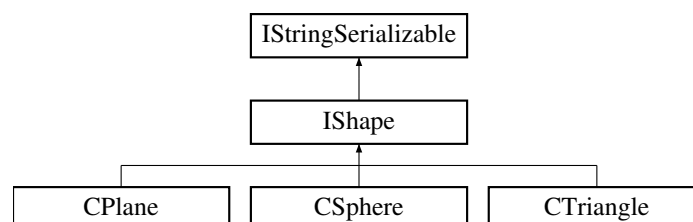
The documentation for this class was generated from the following files:

- E:/dev/VS 14/Projects/raytracer/raytracer/src/[serializable.hpp](#)
- E:/dev/VS 14/Projects/raytracer/raytracer/src/[serializable.cpp](#)

4.15 IShape Class Reference

```
#include <shape.hpp>
```

Inheritance diagram for IShape:



Public Member Functions

- [IShape](#) ([CSharedPointer](#)< [IMaterial](#) > [Material](#))
- virtual [~IShape](#) ()=0
- const [CSharedPointer](#)< [IMaterial](#) > & [GetMaterial](#) () const
- virtual void [GetUV](#) ([Vec3](#) [Point](#), [r32](#) &U, [r32](#) &V) const =0
- virtual bool [Intersect](#) (const [CRay](#) &[Ray](#), [r32](#) tMin, [r32](#) tMax, [SHitInfo](#) &[HitInfo](#)) const =0

Protected Attributes

- [CSharedPointer](#)< [IMaterial](#) > [m_Material](#)

4.15.1 Detailed Description

Generic shape interface.

Definition at line 20 of file `shape.hpp`.

4.15.2 Constructor & Destructor Documentation

4.15.2.1 IShape()

```
IShape::IShape (
    CSharedPointer< IMaterial > Material )
```

Constructor

Parameters

<i>Material</i>	pointer to the shape's material.
-----------------	----------------------------------

Definition at line 3 of file `shape.cpp`.

4.15.2.2 ~IShape()

```
IShape::~~IShape ( ) [pure virtual]
```

Definition at line 9 of file `shape.cpp`.

4.15.3 Member Function Documentation

4.15.3.1 GetMaterial()

```
const CSharedPointer< IMaterial > & IShape::GetMaterial ( ) const
```

Returns the shape's material pointer.

Definition at line 14 of file shape.cpp.

4.15.3.2 GetUV()

```
virtual void IShape::GetUV (
    Vec3 Point,
    r32 & U,
    r32 & V ) const [pure virtual]
```

Returns the UV coordinates of the object at a given point.

Implemented in [CPlane](#), [CSphere](#), and [CTriangle](#).

4.15.3.3 Intersect()

```
virtual bool IShape::Intersect (
    const CRay & Ray,
    r32 tMin,
    r32 tMax,
    SHitInfo & HitInfo ) const [pure virtual]
```

Checks whether a ray intersects with the shape.

Parameters

<i>Ray</i>	the ray to check the intersection with.
<i>tMin</i>	the minimum t parameter of the ray to consider for intersection.
<i>tMax</i>	the maximum t paramter of the ray to considered for intersection.
<i>HitInfo</i>	reference to the object which will store the collision information.

Implemented in [CPlane](#), [CSphere](#), and [CTriangle](#).

4.15.4 Member Data Documentation

4.15.4.1 m_Material

```
CSharedPointer<IMaterial> IShape::m_Material [protected]
```

Pointer to the shape's material

Definition at line 45 of file shape.hpp.

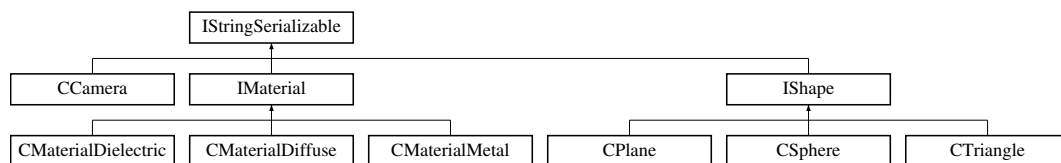
The documentation for this class was generated from the following files:

- E:/dev/VS 14/Projects/raytracer/raytracer/src/[shape.hpp](#)
- E:/dev/VS 14/Projects/raytracer/raytracer/src/[shape.cpp](#)

4.16 IStringSerializable Class Reference

```
#include <serializable.hpp>
```

Inheritance diagram for IStringSerializable:



Public Member Functions

- virtual [~IStringSerializable](#) ()
- virtual std::string & [ReadFromString](#) (std::string &String)=0
- virtual void [WriteToString](#) (std::string &String) const =0

4.16.1 Detailed Description

Serializable interface that supports reading from and writing to strings

Definition at line 23 of file serializable.hpp.

4.16.2 Constructor & Destructor Documentation

4.16.2.1 ~IStringSerializable()

```
IStringSerializable::~~IStringSerializable ( ) [virtual]
```

Definition at line 17 of file serializable.cpp.

4.16.3 Member Function Documentation

4.16.3.1 ReadFromString()

```
virtual std::string& IStringSerializable::ReadFromString (
    std::string & String ) [pure virtual]
```

Reads the object from a string, possibly removing contents from the string.

Implemented in [CMaterialDielectric](#), [CMaterialMetal](#), [CMaterialDiffuse](#), [CCamera](#), [CPlane](#), [CSphere](#), and [CTriangle](#).

4.16.3.2 WriteToString()

```
virtual void IStringSerializable::WriteToString (
    std::string & String ) const [pure virtual]
```

Writes the object to a string.

Implemented in [CMaterialDielectric](#), [CMaterialMetal](#), [CMaterialDiffuse](#), [CCamera](#), [CPlane](#), [CSphere](#), and [CTriangle](#).

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

- E:/dev/VS 14/Projects/raytracer/raytracer/src/[serializable.hpp](#)
- E:/dev/VS 14/Projects/raytracer/raytracer/src/[serializable.cpp](#)

4.17 SArguments Struct Reference

Public Attributes

- std::string [OutputName](#)
- std::string [ScenePath](#)
- s32 [RenderWidth](#)
- s32 [RenderHeight](#)
- u32 [SampleCount](#)
- u32 [MaxDepth](#)
- u32 [MaxThreadCount](#)

4.17.1 Detailed Description

Contains all the possible arguments the program can start with

Definition at line 26 of file raytracer.cpp.

4.17.2 Member Data Documentation

4.17.2.1 MaxDepth

`u32 SArguments::MaxDepth`

Maximum number of times a ray can bounce.

Definition at line 33 of file raytracer.cpp.

4.17.2.2 MaxThreadCount

`u32 SArguments::MaxThreadCount`

Maximum number of threads the app is allowed to create.

Definition at line 34 of file raytracer.cpp.

4.17.2.3 OutputName

`std::string SArguments::OutputName`

The path of the output image file.

Definition at line 28 of file raytracer.cpp.

4.17.2.4 RenderHeight

`s32 SArguments::RenderHeight`

Height of the output image.

Definition at line 31 of file raytracer.cpp.

4.17.2.5 RenderWidth

`s32 SArguments::RenderWidth`

Width of the output image.

Definition at line 30 of file raytracer.cpp.

4.17.2.6 SampleCount

`u32 SArguments::SampleCount`

Number of rays to shoot per pixel.

Definition at line 32 of file raytracer.cpp.

4.17.2.7 ScenePath

`std::string SArguments::ScenePath`

Path to the scene to load (json format).

Definition at line 29 of file raytracer.cpp.

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

- [E:/dev/VS 14/Projects/raytracer/raytracer/src/raytracer.cpp](#)

4.18 SBitmapFileHeader Struct Reference

```
#include <image.hpp>
```

Public Attributes

- [u16 Type](#)
- [u32 Size](#)
- [u16 Reserved1](#)
- [u16 Reserved2](#)
- [u32 Offset](#)

4.18.1 Detailed Description

Struct which holds the .bmp file information. See <https://docs.microsoft.com/en-us/windows/desktop/api/wi>

Definition at line 11 of file image.hpp.

4.18.2 Member Data Documentation

4.18.2.1 Offset

```
u32 SBitmapFileHeader::Offset
```

Definition at line 17 of file image.hpp.

4.18.2.2 Reserved1

```
u16 SBitmapFileHeader::Reserved1
```

Definition at line 15 of file image.hpp.

4.18.2.3 Reserved2

```
u16 SBitmapFileHeader::Reserved2
```

Definition at line 16 of file image.hpp.

4.18.2.4 Size

```
u32 SBitmapFileHeader::Size
```

Definition at line 14 of file image.hpp.

4.18.2.5 Type

```
u16 SBitmapFileHeader::Type
```

Definition at line 13 of file image.hpp.

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

- E:/dev/VS 14/Projects/raytracer/raytracer/src/[image.hpp](#)

4.19 SBitmapInfoHeader Struct Reference

```
#include <image.hpp>
```

Public Attributes

- [u32 Size](#)
- [s32 Width](#)
- [s32 Height](#)
- [u16 Planes](#)
- [u16 BitCount](#)
- [u32 Compression](#)
- [u32 ImageSize](#)
- [s32 PixelsPerMeterX](#)
- [s32 PixelsPerMeterY](#)
- [u32 ClrUsed](#)
- [u32 ClrImportant](#)

4.19.1 Detailed Description

Struct which holds the .bmp image information. See <https://docs.microsoft.com/en-us/windows/desktop/api/>

Definition at line 23 of file image.hpp.

4.19.2 Member Data Documentation

4.19.2.1 BitCount

`u16 SBitmapInfoHeader::BitCount`

Definition at line 29 of file image.hpp.

4.19.2.2 ClrImportant

`u32 SBitmapInfoHeader::ClrImportant`

Definition at line 35 of file image.hpp.

4.19.2.3 ClrUsed

`u32 SBitmapInfoHeader::ClrUsed`

Definition at line 34 of file image.hpp.

4.19.2.4 Compression

`u32 SBitmapInfoHeader::Compression`

Definition at line 30 of file image.hpp.

4.19.2.5 Height

`s32 SBitmapInfoHeader::Height`

Definition at line 27 of file image.hpp.

4.19.2.6 ImageSize

`u32 SBitmapInfoHeader::ImageSize`

Definition at line 31 of file image.hpp.

4.19.2.7 PixelsPerMeterX

`s32 SBitmapInfoHeader::PixelsPerMeterX`

Definition at line 32 of file image.hpp.

4.19.2.8 PixelsPerMeterY

`s32 SBitmapInfoHeader::PixelsPerMeterY`

Definition at line 33 of file image.hpp.

4.19.2.9 Planes

`u16 SBitmapInfoHeader::Planes`

Definition at line 28 of file image.hpp.

4.19.2.10 Size

`u32 SBitmapInfoHeader::Size`

Definition at line 25 of file `image.hpp`.

4.19.2.11 Width

`s32 SBitmapInfoHeader::Width`

Definition at line 26 of file `image.hpp`.

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

- `E:/dev/VS 14/Projects/raytracer/raytracer/src/image.hpp`

4.20 SHitInfo Struct Reference

```
#include <shape.hpp>
```

Public Attributes

- `r32 tVal`
- `Vec3 Point`
- `Vec3 Normal`
- `const IShape * Shape`

4.20.1 Detailed Description

Stores the hit information of a ray-shape collision.

Definition at line 11 of file `shape.hpp`.

4.20.2 Member Data Documentation

4.20.2.1 Normal

`Vec3 SHitInfo::Normal`

The surface normal of the object where the collision occurred.

Definition at line 15 of file `shape.hpp`.

4.20.2.2 Point

```
Vec3 SHitInfo::Point
```

The world coordinate where the collision occurred.

Definition at line 14 of file shape.hpp.

4.20.2.3 Shape

```
const IShape* SHitInfo::Shape
```

Pointer to the object which the ray collided with.

Definition at line 16 of file shape.hpp.

4.20.2.4 tVal

```
r32 SHitInfo::tVal
```

The t parameter of the ray where the collision occurred.

Definition at line 13 of file shape.hpp.

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

- E:/dev/VS 14/Projects/raytracer/raytracer/src/[shape.hpp](#)

4.21 SRegion Struct Reference

```
#include <scene.hpp>
```

Public Attributes

- [CImage Image](#)
- [s32 OffsetX](#)
- [s32 OffsetY](#)

4.21.1 Detailed Description

A region of the final render.

Definition at line 13 of file scene.hpp.

4.21.2 Member Data Documentation

4.21.2.1 Image

`CImage SRegion::Image`

The image of the region.

Definition at line 15 of file scene.hpp.

4.21.2.2 OffsetX

`s32 SRegion::OffsetX`

Horizontal coordinate in the final image.

Definition at line 16 of file scene.hpp.

4.21.2.3 OffsetY

`s32 SRegion::OffsetY`

Vertical coordinate in the final image.

Definition at line 17 of file scene.hpp.

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

- `E:/dev/VS 14/Projects/raytracer/raytracer/src/scene.hpp`

4.22 SRenderParams Struct Reference

```
#include <scene.hpp>
```

Public Attributes

- `u32 SampleCount`
- `u32 MaxDepth`
- `u32 MaxThreadCount`
- `r32 AspectRatio`
- `s32 FullRenderWidth`
- `s32 FullRenderHeight`

4.22.1 Detailed Description

The parameters of the render.

Definition at line 21 of file scene.hpp.

4.22.2 Member Data Documentation

4.22.2.1 AspectRatio

```
r32 SRenderParams::AspectRatio
```

The aspect ratio of the output image. Internal use.

Definition at line 28 of file scene.hpp.

4.22.2.2 FullRenderHeight

```
s32 SRenderParams::FullRenderHeight
```

The height of the final image. Internal use.

Definition at line 30 of file scene.hpp.

4.22.2.3 FullRenderWidth

```
s32 SRenderParams::FullRenderWidth
```

The width of the final image. Internal use.

Definition at line 29 of file scene.hpp.

4.22.2.4 MaxDepth

```
u32 SRenderParams::MaxDepth
```

The maximum number of times a ray can bounce.

Definition at line 24 of file scene.hpp.

4.22.2.5 MaxThreadCount

`u32 SRenderParams::MaxThreadCount`

The maximum number of threads the app can create.

Definition at line 25 of file scene.hpp.

4.22.2.6 SampleCount

`u32 SRenderParams::SampleCount`

The number of pixels to shoot per pixel.

Definition at line 23 of file scene.hpp.

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

- [E:/dev/VS 14/Projects/raytracer/raytracer/src/scene.hpp](#)

4.23 SSharedRenderData Struct Reference

```
#include <scene.hpp>
```

Public Attributes

- `std::atomic<u32>` [PixelsProcessed](#)
- `std::mutex` [PrintMutex](#)

4.23.1 Detailed Description

Multithread data of the render.

Definition at line 34 of file scene.hpp.

4.23.2 Member Data Documentation

4.23.2.1 PixelsProcessed

`std::atomic<u32> SSharedRenderData::PixelsProcessed`

The number of pixels processed by the threads.

Definition at line 36 of file scene.hpp.

4.23.2.2 PrintMutex

```
std::mutex SSharedRenderData::PrintMutex
```

Lock for the output stream.

Definition at line 37 of file scene.hpp.

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

- E:/dev/VS 14/Projects/raytracer/raytracer/src/[scene.hpp](#)

4.24 UColor Union Reference

```
#include <color.hpp>
```

Public Attributes

- [u32 Color](#)
- struct {
 - [u8 Alpha](#)
 - [u8 Blue](#)
 - [u8 Green](#)
 - [u8 Red](#)
- [Components](#)

4.24.1 Detailed Description

Stores color information in 0xRRGGBBAA format.

Definition at line 6 of file color.hpp.

4.24.2 Member Data Documentation

4.24.2.1 Alpha

```
u8 UColor::Alpha
```

Alpha channel of the color.

Definition at line 12 of file color.hpp.

4.24.2.2 Blue

```
u8 UColor::Blue
```

Blue channel of the color.

Definition at line 13 of file color.hpp.

4.24.2.3 Color

```
u32 UColor::Color
```

32 bit unsigned integer holding the color information

Definition at line 8 of file color.hpp.

4.24.2.4 Components

```
struct { ... } UColor::Components
```

Struct to access the components separately.

4.24.2.5 Green

```
u8 UColor::Green
```

Green channel of the color.

Definition at line 14 of file color.hpp.

4.24.2.6 Red

```
u8 UColor::Red
```

Red channel of the color.

Definition at line 15 of file color.hpp.

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

- E:/dev/VS 14/Projects/raytracer/raytracer/src/[color.hpp](#)

4.25 Vec3 Struct Reference

```
#include <common.hpp>
```

Public Member Functions

- [Vec3](#) ()
- [Vec3](#) (r32 X, r32 Y, r32 Z)
- [Vec3 operator-](#) () const
- bool [operator==](#) (const [Vec3](#) &Other) const
- [Vec3](#) & [operator+=](#) (const [Vec3](#) &Other)
- [Vec3](#) & [operator-=](#) (const [Vec3](#) &Other)
- [Vec3](#) & [operator *=](#) (const [Vec3](#) &Other)
- [Vec3 operator+](#) (const [Vec3](#) &Other) const
- [Vec3 operator-](#) (const [Vec3](#) &Other) const
- [Vec3 operator *](#) (const [Vec3](#) &Other) const
- [Vec3](#) & [operator *=](#) (r32 S)
- [Vec3](#) & [operator/=](#) (r32 S)
- [Vec3 operator *](#) (r32 S) const
- [Vec3 operator/](#) (r32 S) const
- [r32 LengthSq](#) () const
- [r32 Length](#) () const

Public Attributes

- [r32 X](#)
- [r32 Y](#)
- [r32 Z](#)

4.25.1 Detailed Description

Mathematical 3D vector.

Definition at line 55 of file common.hpp.

4.25.2 Constructor & Destructor Documentation

4.25.2.1 [Vec3\(\)](#) [1/2]

```
Vec3::Vec3 ( )
```

Empty constructor. Initializes to 0.

Definition at line 70 of file common.cpp.

4.25.2.2 Vec3() [2/2]

```
Vec3::Vec3 (
    r32 X,
    r32 Y,
    r32 Z )
```

Sets the coordinates to the appropriate parameters.

Definition at line 77 of file common.cpp.

4.25.3 Member Function Documentation

4.25.3.1 Length()

```
r32 Vec3::Length ( ) const
```

Returns the Pythagorean length.

Definition at line 186 of file common.cpp.

4.25.3.2 LengthSq()

```
r32 Vec3::LengthSq ( ) const
```

Returns the square of the Pythagorean length.

Definition at line 181 of file common.cpp.

4.25.3.3 operator*() [1/2]

```
Vec3 Vec3::operator * (
    const Vec3 & Other ) const
```

Multiplies a vector with another (component-wise).

Definition at line 121 of file common.cpp.

4.25.3.4 operator*() [2/2]

```
Vec3 Vec3::operator * (
    r32 S ) const
```

Multiplies a vector with a scalar.

Definition at line 163 of file common.cpp.

4.25.3.5 operator*=() [1/2]

```
Vec3 & Vec3::operator *= (
    const Vec3 & Other )
```

Multiplies a vector with another (component-wise).

Definition at line 113 of file common.cpp.

4.25.3.6 operator*=() [2/2]

```
Vec3 & Vec3::operator *= (
    r32 S )
```

Multiplies a vector with a scalar.

Definition at line 142 of file common.cpp.

4.25.3.7 operator+()

```
Vec3 Vec3::operator+ (
    const Vec3 & Other ) const
```

Adds a vector to another (component-wise).

Definition at line 128 of file common.cpp.

4.25.3.8 operator+=()

```
Vec3 & Vec3::operator+= (
    const Vec3 & Other )
```

Adds a vector to another (component-wise).

Definition at line 97 of file common.cpp.

4.25.3.9 operator-() [1/2]

```
Vec3 Vec3::operator- ( ) const
```

Returns the negated version of the vector.

Definition at line 84 of file common.cpp.

4.25.3.10 operator-() [2/2]

```
Vec3 Vec3::operator- (
    const Vec3 & Other ) const
```

Subtracts a vector from another (component-wise).

Definition at line 135 of file common.cpp.

4.25.3.11 operator-=()

```
Vec3 & Vec3::operator-= (
    const Vec3 & Other )
```

Subtracts a vector from another (component-wise).

Definition at line 105 of file common.cpp.

4.25.3.12 operator/()

```
Vec3 Vec3::operator/ (
    r32 S ) const
```

Divides a vector by a scalar.

Definition at line 169 of file common.cpp.

4.25.3.13 operator/=()

```
Vec3 & Vec3::operator/= (
    r32 S )
```

Divides a vector by a scalar.

Definition at line 150 of file common.cpp.

4.25.3.14 operator==()

```
bool Vec3::operator== (
    const Vec3 & Other ) const
```

Checks if all coordinates are equal.

Definition at line 89 of file common.cpp.

4.25.4 Member Data Documentation

4.25.4.1 X

```
r32 Vec3::X
```

X coordinate of the vector.

Definition at line 57 of file common.hpp.

4.25.4.2 Y

```
r32 Vec3::Y
```

Y coordinate of the vector.

Definition at line 58 of file common.hpp.

4.25.4.3 Z

```
r32 Vec3::Z
```

Z coordinate of the vector.

Definition at line 59 of file common.hpp.

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

- E:/dev/VS 14/Projects/raytracer/raytracer/src/[common.hpp](#)
- E:/dev/VS 14/Projects/raytracer/raytracer/src/[common.cpp](#)

Chapter 5

File Documentation

5.1 E:/dev/VS 14/Projects/raytracer/raytracer/src/camera.cpp File Reference

```
#include "camera.hpp"
```

5.2 E:/dev/VS 14/Projects/raytracer/raytracer/src/camera.hpp File Reference

```
#include "common.hpp"  
#include "ray.hpp"  
#include "serializable.hpp"
```

Classes

- class [CCamera](#)

5.3 E:/dev/VS 14/Projects/raytracer/raytracer/src/color.cpp File Reference

```
#include "color.hpp"
```

Functions

- [u32 RGBToU32](#) (u32 R, u32 G, u32 B)
- [u32 RGBAToU32](#) (u32 R, u32 G, u32 B, u32 A)
- [u32 Vec3ToU32](#) (const [Vec3](#) &V)
- [Vec3 U32ToVec3](#) (u32 Color)

5.3.1 Function Documentation

5.3.1.1 RGBAToU32()

```
u32 RGBAToU32 (
    u32 R,
    u32 G,
    u32 B,
    u32 A )
```

Combines RGBA color values to a single u32

Definition at line 12 of file color.cpp.

5.3.1.2 RGBToU32()

```
u32 RGBToU32 (
    u32 R,
    u32 G,
    u32 B )
```

Combines RGB color values to a single u32.

Definition at line 3 of file color.cpp.

5.3.1.3 U32ToVec3()

```
Vec3 U32ToVec3 (
    u32 Color )
```

Converts a color stored in a u32 to a normalized color vector.

Definition at line 26 of file color.cpp.

5.3.1.4 Vec3ToU32()

```
u32 Vec3ToU32 (
    const Vec3 & V )
```

Converts a normalized color (values rangin from 0-1) to a single u32.

Definition at line 21 of file color.cpp.

5.4 E:/dev/VS 14/Projects/raytracer/raytracer/src/color.hpp File Reference

```
#include "common.hpp"
```

Classes

- union [UColor](#)

Functions

- [u32 RGBToU32](#) ([u32](#) R, [u32](#) G, [u32](#) B)
- [u32 RGBAToU32](#) ([u32](#) R, [u32](#) G, [u32](#) B, [u32](#) A)
- [u32 Vec3ToU32](#) (const [Vec3](#) &V)
- [Vec3 U32ToVec3](#) ([u32](#) Color)

5.4.1 Function Documentation

5.4.1.1 RGBAToU32()

```
u32 RGBAToU32 (
    u32 R,
    u32 G,
    u32 B,
    u32 A )
```

Combines RGBA color values to a single [u32](#)

Definition at line 12 of file color.cpp.

5.4.1.2 RGBToU32()

```
u32 RGBToU32 (
    u32 R,
    u32 G,
    u32 B )
```

Combines RGB color values to a single [u32](#).

Definition at line 3 of file color.cpp.

5.4.1.3 U32ToVec3()

```
Vec3 U32ToVec3 (
    u32 Color )
```

Converts a color stored in a u32 to a normalized color vector.

Definition at line 26 of file color.cpp.

5.4.1.4 Vec3ToU32()

```
u32 Vec3ToU32 (
    const Vec3 & V )
```

Converts a normalized color (values rangin from 0-1) to a single u32.

Definition at line 21 of file color.cpp.

5.5 E:/dev/VS 14/Projects/raytracer/raytracer/src/common.cpp File Reference

```
#include "common.hpp"
```

Functions

- [r32 DegreeToRadian](#) (r32 Degree)
- [r32 RadianToDegree](#) (r32 Radian)
- [r32 RandomNormalized](#) ()
- [r32 RandomNormalizedNeg](#) ()
- [std::string ExtractQuote](#) (std::string &String)
- [std::string ExtractBraceContents](#) (std::string &String)
- [Vec3 ExtractVec3](#) (std::string String)
- [void WriteVec3](#) (std::string &String, [Vec3](#) V)
- [Vec3 operator *](#) (r32 S, const [Vec3](#) &V)
- [Vec3 Normalize](#) (const [Vec3](#) &V)
- [r32 Dot](#) (const [Vec3](#) &A, const [Vec3](#) &B)
- [Vec3 Cross](#) (const [Vec3](#) &A, const [Vec3](#) &B)
- [Vec3 Project](#) (const [Vec3](#) &A, const [Vec3](#) &B)
- [Vec3 Reject](#) (const [Vec3](#) &A, const [Vec3](#) &B)
- [Vec3 Reflect](#) ([Vec3](#) Incident, [Vec3](#) Normal)
- [Vec3 Refract](#) ([Vec3](#) Incident, [Vec3](#) Normal, r32 RefractiveRatio)
- [Vec3 Lerp](#) (const [Vec3](#) &A, const [Vec3](#) &B, r32 t)
- [Vec3 RandomInUnitSphere](#) ()

5.5.1 Function Documentation

5.5.1.1 Cross()

```
Vec3 Cross (
    const Vec3 & A,
    const Vec3 & B )
```

Returns two vectors' cross product.

Definition at line 207 of file common.cpp.

5.5.1.2 DegreeToRadian()

```
r32 DegreeToRadian (
    r32 Degree )
```

Converts degrees to radians.

Definition at line 3 of file common.cpp.

5.5.1.3 Dot()

```
r32 Dot (
    const Vec3 & A,
    const Vec3 & B )
```

Returns two vectors' dot product.

Definition at line 202 of file common.cpp.

5.5.1.4 ExtractBraceContents()

```
std::string ExtractBraceContents (
    std::string & String )
```

Extracts content from a string between curly {} braces. The braces and the characters between are removed from the string.

Definition at line 33 of file common.cpp.

5.5.1.5 ExtractQuote()

```
std::string ExtractQuote (
    std::string & String )
```

Extracts a quote from a string. Extracts characters between "" characters, removing them from the string. Quotes are also removed.

Definition at line 25 of file common.cpp.

5.5.1.6 ExtractVec3()

```
Vec3 ExtractVec3 (
    std::string String )
```

Returns a [Vec3](#) from a string. String must be {x, y, z} format.

Definition at line 51 of file common.cpp.

5.5.1.7 Lerp()

```
Vec3 Lerp (
    const Vec3 & A,
    const Vec3 & B,
    r32 t )
```

Linearly interpolates between two vectors given a t value.

Definition at line 242 of file common.cpp.

5.5.1.8 Normalize()

```
Vec3 Normalize (
    const Vec3 & V )
```

Returns the vector divided by its length.

Definition at line 191 of file common.cpp.

5.5.1.9 operator *()

```
Vec3 operator * (
    r32 S,
    const Vec3 & V )
```

Multiplies a vector with a scalar.

Definition at line 176 of file common.cpp.

5.5.1.10 Project()

```
Vec3 Project (
    const Vec3 & A,
    const Vec3 & B )
```

Returns vector A's projection to B.

Definition at line 214 of file common.cpp.

5.5.1.11 RadianToDegree()

```
r32 RadianToDegree (
    r32 Radian )
```

Converts radians to degrees.

Definition at line 8 of file common.cpp.

5.5.1.12 RandomInUnitSphere()

```
Vec3 RandomInUnitSphere ( )
```

Returns a unit vector pointing in a random direction

Definition at line 247 of file common.cpp.

5.5.1.13 RandomNormalized()

```
r32 RandomNormalized ( )
```

Returns a random number between [0..1).

Definition at line 13 of file common.cpp.

5.5.1.14 RandomNormalizedNeg()

```
r32 RandomNormalizedNeg ( )
```

Returns a random number between (-1..1).

Definition at line 20 of file common.cpp.

5.5.1.15 Reflect()

```
Vec3 Reflect (
    Vec3 Incident,
    Vec3 Normal )
```

Returns a vector's reflection given the surface normal.

Definition at line 224 of file common.cpp.

5.5.1.16 Refract()

```
Vec3 Refract (
    Vec3 Incident,
    Vec3 Normal,
    r32 RefractiveRatio )
```

Returns a vector's refraction given a surface normal and the refractive index ratio between the medium. Returns a null vector if no refraction is possible.

Definition at line 229 of file common.cpp.

5.5.1.17 Reject()

```
Vec3 Reject (
    const Vec3 & A,
    const Vec3 & B )
```

Returns vector A's rejection from B.

Definition at line 219 of file common.cpp.

5.5.1.18 WriteVec3()

```
void WriteVec3 (
    std::string & String,
    Vec3 V )
```

Writes a [Vec3](#) to a string in {x, y, z} format.

Definition at line 63 of file common.cpp.

5.6 E:/dev/VS 14/Projects/raytracer/raytracer/src/common.hpp File Reference

```
#include <cinttypes>
#include <cfloat>
#include <cmath>
#include <random>
#include <algorithm>
#include <string>
#include <stdexcept>
#include <iostream>
#include <iomanip>
#include <fstream>
#include <cstring>
#include <atomic>
#include <mutex>
#include <sstream>
```

Classes

- struct [Vec3](#)

Typedefs

- typedef uint8_t [u8](#)
- typedef int8_t [s8](#)
- typedef uint16_t [u16](#)
- typedef int16_t [s16](#)
- typedef uint32_t [u32](#)
- typedef int32_t [s32](#)
- typedef uint64_t [u64](#)
- typedef int64_t [s64](#)
- typedef float [r32](#)
- typedef double [r64](#)

Functions

- [r32 DegreeToRadian](#) ([r32](#) Degree)
- [r32 RadianToDegree](#) ([r32](#) Radian)
- [r32 RandomNormalized](#) ()
- [r32 RandomNormalizedNeg](#) ()
- `template<class T >`
`T Clamp` (T Val, T Min, T Max)
- [Vec3 operator *](#) ([r32](#) S, const [Vec3](#) &V)
- [Vec3 Normalize](#) (const [Vec3](#) &V)
- [r32 Dot](#) (const [Vec3](#) &A, const [Vec3](#) &B)
- [Vec3 Cross](#) (const [Vec3](#) &A, const [Vec3](#) &B)
- [Vec3 Project](#) (const [Vec3](#) &A, const [Vec3](#) &B)
- [Vec3 Reject](#) (const [Vec3](#) &A, const [Vec3](#) &B)
- [Vec3 Reflect](#) ([Vec3](#) Incident, [Vec3](#) Normal)
- [Vec3 Refract](#) ([Vec3](#) Incident, [Vec3](#) Normal, [r32](#) RefractiveRatio)
- [Vec3 Lerp](#) (const [Vec3](#) &A, const [Vec3](#) &B, [r32](#) t)
- [Vec3 RandomInUnitSphere](#) ()
- `std::string` [ExtractQuote](#) (`std::string` &String)
- `std::string` [ExtractBraceContents](#) (`std::string` &String)
- [Vec3 ExtractVec3](#) (`std::string` String)
- `void` [WriteVec3](#) (`std::string` &String, [Vec3](#) V)

Variables

- `const` [r32 Pi32](#) = 3.1415927f

5.6.1 Typedef Documentation

5.6.1.1 r32

```
typedef float r32
```

32 bit floating point value.

Definition at line 27 of file common.hpp.

5.6.1.2 r64

```
typedef double r64
```

64 bit floating point value.

Definition at line 28 of file common.hpp.

5.6.1.3 s16

```
typedef int16_t s16
```

Signed 16 bit integer.

Definition at line 21 of file common.hpp.

5.6.1.4 s32

```
typedef int32_t s32
```

Signed 32 bit integer.

Definition at line 23 of file common.hpp.

5.6.1.5 s64

```
typedef int64_t s64
```

Signed 64 bit integer.

Definition at line 25 of file common.hpp.

5.6.1.6 s8

```
typedef int8_t s8
```

Signed 8 bit integer.

Definition at line 19 of file common.hpp.

5.6.1.7 u16

```
typedef uint16_t u16
```

Unsigned 16 bit integer.

Definition at line 20 of file common.hpp.

5.6.1.8 u32

```
typedef uint32_t u32
```

Unsigned 32 bit integer.

Definition at line 22 of file common.hpp.

5.6.1.9 u64

```
typedef uint64_t u64
```

Unsigned 64 bit integer.

Definition at line 24 of file common.hpp.

5.6.1.10 u8

```
typedef uint8_t u8
```

Unsigned 8 bit integer.

Definition at line 18 of file common.hpp.

5.6.2 Function Documentation

5.6.2.1 Clamp()

```
template<class T >  
T Clamp (  
    T Val,  
    T Min,  
    T Max )
```

Clamps a value between a range .

Parameters

<i>Val</i>	value to clamp.
<i>Min</i>	lower bound of the clamp.
<i>Max</i>	upper bound of the clamp.

Definition at line 48 of file common.hpp.

5.6.2.2 Cross()

```
Vec3 Cross (
    const Vec3 & A,
    const Vec3 & B )
```

Returns two vectors' cross product.

Definition at line 207 of file common.cpp.

5.6.2.3 DegreeToRadian()

```
r32 DegreeToRadian (
    r32 Degree )
```

Converts degrees to radians.

Definition at line 3 of file common.cpp.

5.6.2.4 Dot()

```
r32 Dot (
    const Vec3 & A,
    const Vec3 & B )
```

Returns two vectors' dot product.

Definition at line 202 of file common.cpp.

5.6.2.5 ExtractBraceContents()

```
std::string ExtractBraceContents (
    std::string & String )
```

Extracts content from a string between curly {} braces. The braces and the characters between are removed from the string.

Definition at line 33 of file common.cpp.

5.6.2.6 ExtractQuote()

```
std::string ExtractQuote (
    std::string & String )
```

Extracts a quote from a string. Extracts characters between "" characters, removing them from the string. Quotes are also removed.

Definition at line 25 of file common.cpp.

5.6.2.7 ExtractVec3()

```
Vec3 ExtractVec3 (
    std::string String )
```

Returns a [Vec3](#) from a string. String must be {x, y, z} format.

Definition at line 51 of file common.cpp.

5.6.2.8 Lerp()

```
Vec3 Lerp (
    const Vec3 & A,
    const Vec3 & B,
    r32 t )
```

Linearly interpolates between to vectors given a t value.

Definition at line 242 of file common.cpp.

5.6.2.9 Normalize()

```
Vec3 Normalize (
    const Vec3 & V )
```

Returns the vector divided by its length.

Definition at line 191 of file common.cpp.

5.6.2.10 operator*()

```
Vec3 operator * (
    r32 S,
    const Vec3 & V )
```

Multiplies a vector with a scalar.

Definition at line 176 of file common.cpp.

5.6.2.11 Project()

```
Vec3 Project (
    const Vec3 & A,
    const Vec3 & B )
```

Returns vector A's projection to B.

Definition at line 214 of file common.cpp.

5.6.2.12 RadianToDegree()

```
r32 RadianToDegree (
    r32 Radian )
```

Converts radians to degrees.

Definition at line 8 of file common.cpp.

5.6.2.13 RandomInUnitSphere()

```
Vec3 RandomInUnitSphere ( )
```

Returns a unit vector pointing in a random direction

Definition at line 247 of file common.cpp.

5.6.2.14 RandomNormalized()

```
r32 RandomNormalized ( )
```

Returns a random number between [0..1).

Definition at line 13 of file common.cpp.

5.6.2.15 RandomNormalizedNeg()

```
r32 RandomNormalizedNeg ( )
```

Returns a random number between (-1..1).

Definition at line 20 of file common.cpp.

5.6.2.16 Reflect()

```
Vec3 Reflect (
    Vec3 Incident,
    Vec3 Normal )
```

Returns a vector's reflection given the surface normal.

Definition at line 224 of file common.cpp.

5.6.2.17 Refract()

```
Vec3 Refract (
    Vec3 Incident,
    Vec3 Normal,
    r32 RefractiveRatio )
```

Returns a vector's refraction given a surface normal and the refractive index ratio between the medium. Returns a null vector if no refraction is possible.

Definition at line 229 of file common.cpp.

5.6.2.18 Reject()

```
Vec3 Reject (
    const Vec3 & A,
    const Vec3 & B )
```

Returns vector A's rejection from B.

Definition at line 219 of file common.cpp.

5.6.2.19 WriteVec3()

```
void WriteVec3 (
    std::string & String,
    Vec3 V )
```

Writes a [Vec3](#) to a string in {x, y, z} format.

Definition at line 63 of file common.cpp.

5.6.3 Variable Documentation

5.6.3.1 Pi32

```
const r32 Pi32 = 3.1415927f
```

32 bit constant for pi.

Definition at line 30 of file common.hpp.

5.7 E:/dev/VS 14/Projects/raytracer/raytracer/src/heterostore.hpp File Reference

```
#include <algorithm>
#include "sharedpointer.hpp"
```

Classes

- class [CHeteroStore< T >](#)

5.8 E:/dev/VS 14/Projects/raytracer/raytracer/src/image.cpp File Reference

```
#include "image.hpp"
#include <stdexcept>
```

5.9 E:/dev/VS 14/Projects/raytracer/raytracer/src/image.hpp File Reference

```
#include "common.hpp"
#include "color.hpp"
#include "serializable.hpp"
```

Classes

- struct [SBitmapFileHeader](#)
- struct [SBitmapInfoHeader](#)
- class [CImage](#)

5.10 E:/dev/VS 14/Projects/raytracer/raytracer/src/material.cpp File Reference

```
#include "material.hpp"
```

5.11 E:/dev/VS 14/Projects/raytracer/raytracer/src/material.hpp File Reference

```
#include "common.hpp"  
#include "sharedpointer.hpp"  
#include "ray.hpp"  
#include "image.hpp"
```

Classes

- class [IMaterial](#)
- class [CMaterialDiffuse](#)
- class [CMaterialMetal](#)
- class [CMaterialDielectric](#)

5.12 E:/dev/VS 14/Projects/raytracer/raytracer/src/plane.cpp File Reference

```
#include "plane.hpp"
```

5.13 E:/dev/VS 14/Projects/raytracer/raytracer/src/plane.hpp File Reference

```
#include "shape.hpp"
```

Classes

- class [CPlane](#)

5.14 E:/dev/VS 14/Projects/raytracer/raytracer/src/ray.cpp File Reference

```
#include "ray.hpp"
```


5.15 E:/dev/VS 14/Projects/raytracer/raytracer/src/ray.hpp File Reference

```
#include "common.hpp"
```

Classes

- class [CRay](#)

5.16 E:/dev/VS 14/Projects/raytracer/raytracer/src/raytracer.cpp File Reference

```
#include "common.hpp"  
#include "image.hpp"  
#include "shape.hpp"  
#include "ray.hpp"  
#include "camera.hpp"  
#include "scene.hpp"  
#include "sphere.hpp"  
#include "plane.hpp"
```

Classes

- struct [SArguments](#)

Functions

- void [PrintHelp](#) ()
- bool [ParseArguments](#) (int argc, char **argv, [SArguments](#) &Arguments)
- int [main](#) (int argc, char **argv)

5.16.1 Function Documentation

5.16.1.1 main()

```
int main (  
    int argc,  
    char ** argv )
```

Definition at line 132 of file raytracer.cpp.

5.16.1.2 ParseArguments()

```
bool ParseArguments (
    int argc,
    char ** argv,
    SArguments & Arguments )
```

Parses the command line arguments.

Definition at line 39 of file raytracer.cpp.

5.16.1.3 PrintHelp()

```
void PrintHelp ( )
```

Displays the command line usage for the app.

Definition at line 13 of file raytracer.cpp.

5.17 E:/dev/VS 14/Projects/raytracer/raytracer/src/scene.cpp File Reference

```
#include "scene.hpp"
#include "common.hpp"
#include <iostream>
#include <iomanip>
#include <thread>
#include <chrono>
```

5.18 E:/dev/VS 14/Projects/raytracer/raytracer/src/scene.hpp File Reference

```
#include "heterostore.hpp"
#include "ray.hpp"
#include "image.hpp"
#include "camera.hpp"
#include "shape.hpp"
#include "plane.hpp"
#include "sphere.hpp"
```

Classes

- struct [SRegion](#)
- struct [SRenderParams](#)
- struct [SSharedRenderData](#)
- class [CScene](#)

5.19 E:/dev/VS 14/Projects/raytracer/raytracer/src/serializable.cpp File Reference

```
#include "serializable.hpp"
```

Functions

- `std::istream & operator>>` (`std::istream &Stream`, `ISerializable *Var`)
- `std::ostream & operator<<` (`std::ostream &Stream`, `const ISerializable *Var`)

5.19.1 Function Documentation

5.19.1.1 `operator<<()`

```
std::ostream& operator<< (  
    std::ostream & Stream,  
    const ISerializable * Var )
```

Stream operator overload to write the object to a stream.

Definition at line 12 of file `serializable.cpp`.

5.19.1.2 `operator>>()`

```
std::istream& operator>> (  
    std::istream & Stream,  
    ISerializable * Var )
```

Stream operator overload to read the object from a stream.

Definition at line 8 of file `serializable.cpp`.

5.20 E:/dev/VS 14/Projects/raytracer/raytracer/src/serializable.hpp File Reference

```
#include "common.hpp"
```

Classes

- class `ISerializable`
- class `IStringSerializable`

Functions

- `std::istream & operator>> (std::istream &Stream, ISerializable *Var)`
- `std::ostream & operator<< (std::ostream &Stream, const ISerializable *Var)`

5.20.1 Function Documentation

5.20.1.1 `operator<<()`

```
std::ostream& operator<< (
    std::ostream & Stream,
    const ISerializable * Var )
```

Stream operator overload to write the object to a stream.

Definition at line 12 of file serializable.cpp.

5.20.1.2 `operator>>()`

```
std::istream& operator>> (
    std::istream & Stream,
    ISerializable * Var )
```

Stream operator overload to read the object from a stream.

Definition at line 8 of file serializable.cpp.

5.21 E:/dev/VS 14/Projects/raytracer/raytracer/src/shape.cpp File Reference

```
#include "shape.hpp"
```

5.22 E:/dev/VS 14/Projects/raytracer/raytracer/src/shape.hpp File Reference

```
#include "common.hpp"
#include "sharedpointer.hpp"
#include "ray.hpp"
#include "material.hpp"
```

Classes

- struct [SHitInfo](#)
- class [IShape](#)

5.23 E:/dev/VS 14/Projects/raytracer/raytracer/src/sharedpointer.hpp File Reference

```
#include <stdexcept>
```

Classes

- class [CSharedPointer< T >](#)

5.24 E:/dev/VS 14/Projects/raytracer/raytracer/src/sphere.cpp File Reference

```
#include "sphere.hpp"
```

5.25 E:/dev/VS 14/Projects/raytracer/raytracer/src/sphere.hpp File Reference

```
#include "shape.hpp"
```

Classes

- class [CSphere](#)

5.26 E:/dev/VS 14/Projects/raytracer/raytracer/src/triangle.cpp File Reference

```
#include "triangle.hpp"
```

5.27 E:/dev/VS 14/Projects/raytracer/raytracer/src/triangle.hpp File Reference

```
#include "shape.hpp"
```

Classes

- class [CTriangle](#)

Index

- ~CHeteroStore
 - CHeteroStore< T >, [11](#)
- ~CImage
 - CImage, [14](#)
- ~CPlane
 - CPlane, [25](#)
- ~CScene
 - CScene, [29](#)
- ~CSharedPointer
 - CSharedPointer< T >, [32](#)
- ~CSphere
 - CSphere, [36](#)
- ~CTriangle
 - CTriangle, [38](#)
- ~ISerializable
 - ISerializable, [44](#)
- ~IShape
 - IShape, [45](#)
- ~IStringSerializable
 - IStringSerializable, [47](#)
- AddShape
 - CScene, [29](#)
- Alpha
 - UColor, [59](#)
- AspectRatio
 - SRenderParams, [57](#)
- BitCount
 - SBitmapInfoHeader, [52](#)
- Blit
 - CImage, [14](#)
- Blue
 - UColor, [59](#)
- CCamera, [7](#)
 - CCamera, [8](#)
 - GetForward, [8](#)
 - GetPosition, [8](#)
 - GetUp, [8](#)
 - RayFromUV, [8](#)
 - ReadFromString, [9](#)
 - SetForward, [9](#)
 - SetPosition, [9](#)
 - SetUp, [9](#)
 - WriteToString, [10](#)
- CHeteroStore
 - CHeteroStore< T >, [11](#)
- CHeteroStore< T >, [10](#)
 - ~CHeteroStore, [11](#)
- CHeteroStore, [11](#)
 - operator[], [11](#)
 - PushBack, [11](#), [12](#)
 - Resize, [12](#)
 - Size, [12](#)
- CImage, [13](#)
 - ~CImage, [14](#)
 - Blit, [14](#)
 - CImage, [13](#), [14](#)
 - Height, [14](#)
 - operator(), [15](#)
 - operator=, [15](#)
 - Pixels, [15](#)
 - Read, [16](#)
 - Resize, [16](#)
 - Sample, [16](#)
 - Width, [17](#)
 - Write, [17](#)
- Clamp
 - common.hpp, [78](#)
- CIsrImportant
 - SBitmapInfoHeader, [52](#)
- CIsrUsed
 - SBitmapInfoHeader, [52](#)
- CMaterialDielectric, [17](#)
 - CMaterialDielectric, [18](#)
 - ReadFromString, [18](#)
 - Scatter, [19](#)
 - WriteToString, [19](#)
- CMaterialDiffuse, [19](#)
 - CMaterialDiffuse, [20](#)
 - ReadFromString, [20](#)
 - Scatter, [20](#)
 - WriteToString, [21](#)
- CMaterialMetal, [21](#)
 - CMaterialMetal, [22](#)
 - m_Fuzziness, [23](#)
 - ReadFromString, [22](#)
 - Scatter, [22](#)
 - WriteToString, [23](#)
- Color
 - UColor, [60](#)
- color.cpp
 - RGBAToU32, [68](#)
 - RGBToU32, [68](#)
 - U32ToVec3, [68](#)
 - Vec3ToU32, [68](#)
- color.hpp
 - RGBAToU32, [69](#)

- RGBToU32, [69](#)
- U32ToVec3, [69](#)
- Vec3ToU32, [70](#)
- common.cpp
 - Cross, [70](#)
 - DegreeToRadian, [71](#)
 - Dot, [71](#)
 - ExtractBraceContents, [71](#)
 - ExtractQuote, [71](#)
 - ExtractVec3, [72](#)
 - Lerp, [72](#)
 - Normalize, [72](#)
 - operator *, [72](#)
 - Project, [73](#)
 - RadianToDegree, [73](#)
 - RandomInUnitSphere, [73](#)
 - RandomNormalized, [73](#)
 - RandomNormalizedNeg, [73](#)
 - Reflect, [74](#)
 - Refract, [74](#)
 - Reject, [74](#)
 - WriteVec3, [74](#)
- common.hpp
 - Clamp, [78](#)
 - Cross, [79](#)
 - DegreeToRadian, [79](#)
 - Dot, [79](#)
 - ExtractBraceContents, [79](#)
 - ExtractQuote, [79](#)
 - ExtractVec3, [80](#)
 - Lerp, [80](#)
 - Normalize, [80](#)
 - operator *, [80](#)
 - Pi32, [83](#)
 - Project, [81](#)
 - r32, [76](#)
 - r64, [76](#)
 - RadianToDegree, [81](#)
 - RandomInUnitSphere, [81](#)
 - RandomNormalized, [81](#)
 - RandomNormalizedNeg, [81](#)
 - Reflect, [82](#)
 - Refract, [82](#)
 - Reject, [82](#)
 - s16, [76](#)
 - s32, [77](#)
 - s64, [77](#)
 - s8, [77](#)
 - u16, [77](#)
 - u32, [77](#)
 - u64, [78](#)
 - u8, [78](#)
 - WriteVec3, [82](#)
- Components
 - UColor, [60](#)
- Compression
 - SBitmapInfoHeader, [52](#)
- CPlane, [24](#)
- ~CPlane, [25](#)
- CPlane, [24](#)
- GetUV, [25](#)
- Intersect, [25](#)
- ReadFromString, [26](#)
- WriteToString, [26](#)
- CRay, [26](#)
- CRay, [27](#)
- Direction, [27](#)
- Origin, [28](#)
- PointAt, [28](#)
- Cross
 - common.cpp, [70](#)
 - common.hpp, [79](#)
- CScene, [28](#)
- ~CScene, [29](#)
- AddShape, [29](#)
- CScene, [29](#)
- Read, [30](#)
- Render, [30](#)
- RenderRegion, [30](#)
- SetCamera, [31](#)
- Write, [31](#)
- CSharedPointer
 - CSharedPointer< T >, [32](#)
- CSharedPointer< T >, [31](#)
- ~CSharedPointer, [32](#)
- CSharedPointer, [32](#)
- IsNull, [33](#)
- operator *, [33](#)
- operator const T *, [33](#)
- operator T *, [33](#)
- operator->, [33](#), [34](#)
- operator=, [34](#)
- Pointer, [34](#)
- CSphere, [35](#)
- ~CSphere, [36](#)
- CSphere, [35](#)
- GetUV, [36](#)
- Intersect, [36](#)
- ReadFromString, [37](#)
- WriteToString, [37](#)
- CTriangle, [37](#)
- ~CTriangle, [38](#)
- CTriangle, [38](#)
- GetUV, [38](#)
- Intersect, [39](#)
- ReadFromString, [39](#)
- WriteToString, [39](#)
- DegreeToRadian
 - common.cpp, [71](#)
 - common.hpp, [79](#)
- Direction
 - CRay, [27](#)
- Dot
 - common.cpp, [71](#)
 - common.hpp, [79](#)

E:/dev/VS 14/Projects/raytracer/raytracer/src/camera.cpp,	ExtractVec3
67	common.cpp, 72
E:/dev/VS 14/Projects/raytracer/raytracer/src/camera.hpp,	common.hpp, 80
67	
E:/dev/VS 14/Projects/raytracer/raytracer/src/color.cpp,	FullRenderHeight
67	SRenderParams, 57
E:/dev/VS 14/Projects/raytracer/raytracer/src/color.hpp,	FullRenderWidth
69	SRenderParams, 57
E:/dev/VS 14/Projects/raytracer/raytracer/src/common.cpp,	GetColor
70	IMaterial, 42
E:/dev/VS 14/Projects/raytracer/raytracer/src/common.hpp,	GetForward
75	CCamera, 8
E:/dev/VS 14/Projects/raytracer/raytracer/src/heterostore.hpp,	GetMaterial
83	IShape, 45
E:/dev/VS 14/Projects/raytracer/raytracer/src/image.cpp,	GetPosition
83	CCamera, 8
E:/dev/VS 14/Projects/raytracer/raytracer/src/image.hpp,	GetTexture
83	IMaterial, 42
E:/dev/VS 14/Projects/raytracer/raytracer/src/material.cpp,	GetUp
84	CCamera, 8
E:/dev/VS 14/Projects/raytracer/raytracer/src/material.hpp,	GetUV
84	CPlane, 25
E:/dev/VS 14/Projects/raytracer/raytracer/src/plane.cpp,	CSphere, 36
84	CTriangle, 38
E:/dev/VS 14/Projects/raytracer/raytracer/src/plane.hpp,	IShape, 46
84	Green
E:/dev/VS 14/Projects/raytracer/raytracer/src/ray.cpp, 84	UColor, 60
E:/dev/VS 14/Projects/raytracer/raytracer/src/ray.hpp, 85	
E:/dev/VS 14/Projects/raytracer/raytracer/src/raytracer.cpp,	Height
85	CImage, 14
E:/dev/VS 14/Projects/raytracer/raytracer/src/scene.cpp,	SBitmapInfoHeader, 53
86	
E:/dev/VS 14/Projects/raytracer/raytracer/src/scene.hpp,	Image
86	SRegion, 56
E:/dev/VS 14/Projects/raytracer/raytracer/src/serializable.cpp,	ImageSize
87	SBitmapInfoHeader, 53
E:/dev/VS 14/Projects/raytracer/raytracer/src/serializable.hpp,	IMaterial, 40
87	GetColor, 42
E:/dev/VS 14/Projects/raytracer/raytracer/src/shape.cpp,	GetTexture, 42
88	IMaterial, 41
E:/dev/VS 14/Projects/raytracer/raytracer/src/shape.hpp,	m_Color, 43
88	m_Texture, 43
E:/dev/VS 14/Projects/raytracer/raytracer/src/sharedpointer.hpp,	Scatter, 42
89	Intersect
E:/dev/VS 14/Projects/raytracer/raytracer/src/sphere.cpp,	CPlane, 25
89	CSphere, 36
E:/dev/VS 14/Projects/raytracer/raytracer/src/sphere.hpp,	CTriangle, 39
89	IShape, 46
E:/dev/VS 14/Projects/raytracer/raytracer/src/triangle.cpp,	ISerializable, 43
89	~ISerializable, 44
E:/dev/VS 14/Projects/raytracer/raytracer/src/triangle.hpp,	Read, 44
89	Write, 44
ExtractBraceContents	IShape, 44
common.cpp, 71	~IShape, 45
common.hpp, 79	GetMaterial, 45
ExtractQuote	GetUV, 46
common.cpp, 71	Intersect, 46
common.hpp, 79	IShape, 45

- m_Material, 46
- IsNull
 - CSharedPointer< T >, 33
- IStringSerializable, 47
 - ~IStringSerializable, 47
 - ReadFromString, 48
 - WriteToString, 48
- Length
 - Vec3, 62
- LengthSq
 - Vec3, 62
- Lerp
 - common.cpp, 72
 - common.hpp, 80
- m_Color
 - IMaterial, 43
- m_Fuzziness
 - CMaterialMetal, 23
- m_Material
 - IShape, 46
- m_Texture
 - IMaterial, 43
- main
 - raytracer.cpp, 85
- MaxDepth
 - SArguments, 49
 - SRenderParams, 57
- MaxThreadCount
 - SArguments, 49
 - SRenderParams, 57
- Normal
 - SHitInfo, 54
- Normalize
 - common.cpp, 72
 - common.hpp, 80
- Offset
 - SBitmapFileHeader, 50
- OffsetX
 - SRegion, 56
- OffsetY
 - SRegion, 56
- operator *
 - common.cpp, 72
 - common.hpp, 80
 - CSharedPointer< T >, 33
 - Vec3, 62
- operator *=
 - Vec3, 63
- operator const T *
 - CSharedPointer< T >, 33
- operator T *
 - CSharedPointer< T >, 33
- operator <<
 - serializable.cpp, 87
 - serializable.hpp, 88
- operator >>
 - serializable.cpp, 87
 - serializable.hpp, 88
- operator()
 - CImage, 15
- operator+
 - Vec3, 63
- operator+=
 - Vec3, 63
- operator-
 - Vec3, 63, 64
- operator->
 - CSharedPointer< T >, 33, 34
- operator=
 - Vec3, 64
- operator/
 - Vec3, 64
- operator/=
 - Vec3, 64
- operator=
 - CImage, 15
 - CSharedPointer< T >, 34
- operator==
 - Vec3, 64
- operator[]
 - CHeteroStore< T >, 11
- Origin
 - CRay, 28
- OutputName
 - SArguments, 49
- ParseArguments
 - raytracer.cpp, 85
- Pi32
 - common.hpp, 83
- Pixels
 - CImage, 15
- PixelsPerMeterX
 - SBitmapInfoHeader, 53
- PixelsPerMeterY
 - SBitmapInfoHeader, 53
- PixelsProcessed
 - SSharedRenderData, 58
- Planes
 - SBitmapInfoHeader, 53
- Point
 - SHitInfo, 54
- PointAt
 - CRay, 28
- Pointer
 - CSharedPointer< T >, 34
- PrintHelp
 - raytracer.cpp, 86
- PrintMutex
 - SSharedRenderData, 58
- Project
 - common.cpp, 73
 - common.hpp, 81
- PushBack

- CHeteroStore< T >, [11](#), [12](#)
- r32
 - common.hpp, [76](#)
- r64
 - common.hpp, [76](#)
- RadianToDegree
 - common.cpp, [73](#)
 - common.hpp, [81](#)
- RandomInUnitSphere
 - common.cpp, [73](#)
 - common.hpp, [81](#)
- RandomNormalized
 - common.cpp, [73](#)
 - common.hpp, [81](#)
- RandomNormalizedNeg
 - common.cpp, [73](#)
 - common.hpp, [81](#)
- RayFromUV
 - CCamera, [8](#)
- raytracer.cpp
 - main, [85](#)
 - ParseArguments, [85](#)
 - PrintHelp, [86](#)
- Read
 - CImage, [16](#)
 - CScene, [30](#)
 - ISerializable, [44](#)
- ReadFromString
 - CCamera, [9](#)
 - CMaterialDielectric, [18](#)
 - CMaterialDiffuse, [20](#)
 - CMaterialMetal, [22](#)
 - CPlane, [26](#)
 - CSphere, [37](#)
 - CTriangle, [39](#)
 - IStringSerializable, [48](#)
- Red
 - UColor, [60](#)
- Reflect
 - common.cpp, [74](#)
 - common.hpp, [82](#)
- Refract
 - common.cpp, [74](#)
 - common.hpp, [82](#)
- Reject
 - common.cpp, [74](#)
 - common.hpp, [82](#)
- Render
 - CScene, [30](#)
- RenderHeight
 - SArguments, [49](#)
- RenderRegion
 - CScene, [30](#)
- RenderWidth
 - SArguments, [49](#)
- Reserved1
 - SBitmapFileHeader, [51](#)
- Reserved2
 - SBitmapFileHeader, [51](#)
- SBitmapFileHeader, [51](#)
- Resize
 - CHeteroStore< T >, [12](#)
 - CImage, [16](#)
- RGBAToU32
 - color.cpp, [68](#)
 - color.hpp, [69](#)
- RGBToU32
 - color.cpp, [68](#)
 - color.hpp, [69](#)
- s16
 - common.hpp, [76](#)
- s32
 - common.hpp, [77](#)
- s64
 - common.hpp, [77](#)
- s8
 - common.hpp, [77](#)
- Sample
 - CImage, [16](#)
- SampleCount
 - SArguments, [49](#)
 - SRenderParams, [58](#)
- SArguments, [48](#)
 - MaxDepth, [49](#)
 - MaxThreadCount, [49](#)
 - OutputName, [49](#)
 - RenderHeight, [49](#)
 - RenderWidth, [49](#)
 - SampleCount, [49](#)
 - ScenePath, [50](#)
- SBitmapFileHeader, [50](#)
 - Offset, [50](#)
 - Reserved1, [51](#)
 - Reserved2, [51](#)
 - Size, [51](#)
 - Type, [51](#)
- SBitmapInfoHeader, [51](#)
 - BitCount, [52](#)
 - ClrImportant, [52](#)
 - ClrUsed, [52](#)
 - Compression, [52](#)
 - Height, [53](#)
 - ImageSize, [53](#)
 - PixelsPerMeterX, [53](#)
 - PixelsPerMeterY, [53](#)
 - Planes, [53](#)
 - Size, [53](#)
 - Width, [54](#)
- Scatter
 - CMaterialDielectric, [19](#)
 - CMaterialDiffuse, [20](#)
 - CMaterialMetal, [22](#)
 - IMaterial, [42](#)
- ScenePath
 - SArguments, [50](#)
- serializable.cpp
 - operator<<, [87](#)

- operator>>, [87](#)
- serializable.hpp
 - operator<<, [88](#)
 - operator>>, [88](#)
- SetCamera
 - CScene, [31](#)
- SetForward
 - CCamera, [9](#)
- SetPosition
 - CCamera, [9](#)
- SetUp
 - CCamera, [9](#)
- Shape
 - SHitInfo, [55](#)
- SHitInfo, [54](#)
 - Normal, [54](#)
 - Point, [54](#)
 - Shape, [55](#)
 - tVal, [55](#)
- Size
 - CHeteroStore< T >, [12](#)
 - SBitmapFileHeader, [51](#)
 - SBitmapInfoHeader, [53](#)
- SRegion, [55](#)
 - Image, [56](#)
 - OffsetX, [56](#)
 - OffsetY, [56](#)
- SRenderParams, [56](#)
 - AspectRatio, [57](#)
 - FullRenderHeight, [57](#)
 - FullRenderWidth, [57](#)
 - MaxDepth, [57](#)
 - MaxThreadCount, [57](#)
 - SampleCount, [58](#)
- SSharedRenderData, [58](#)
 - PixelsProcessed, [58](#)
 - PrintMutex, [58](#)
- tVal
 - SHitInfo, [55](#)
- Type
 - SBitmapFileHeader, [51](#)
- u16
 - common.hpp, [77](#)
- u32
 - common.hpp, [77](#)
- U32ToVec3
 - color.cpp, [68](#)
 - color.hpp, [69](#)
- u64
 - common.hpp, [78](#)
- u8
 - common.hpp, [78](#)
- UColor, [59](#)
 - Alpha, [59](#)
 - Blue, [59](#)
 - Color, [60](#)
 - Components, [60](#)
 - Green, [60](#)
 - Red, [60](#)
- Vec3, [61](#)
 - Length, [62](#)
 - LengthSq, [62](#)
 - operator *, [62](#)
 - operator *=, [63](#)
 - operator+, [63](#)
 - operator+=, [63](#)
 - operator-, [63](#), [64](#)
 - operator-=, [64](#)
 - operator/, [64](#)
 - operator/=: [64](#)
 - operator==, [64](#)
 - Vec3, [61](#)
 - X, [65](#)
 - Y, [65](#)
 - Z, [65](#)
- Vec3ToU32
 - color.cpp, [68](#)
 - color.hpp, [70](#)
- Width
 - CImage, [17](#)
 - SBitmapInfoHeader, [54](#)
- Write
 - CImage, [17](#)
 - CScene, [31](#)
 - ISerializable, [44](#)
- WriteToString
 - CCamera, [10](#)
 - CMaterialDielectric, [19](#)
 - CMaterialDiffuse, [21](#)
 - CMaterialMetal, [23](#)
 - CPlane, [26](#)
 - CSphere, [37](#)
 - CTriangle, [39](#)
 - IStringSerializable, [48](#)
- WriteVec3
 - common.cpp, [74](#)
 - common.hpp, [82](#)
- X
 - Vec3, [65](#)
- Y
 - Vec3, [65](#)
- Z
 - Vec3, [65](#)