OpenGL Cube Game

Generated by Doxygen 1.8.9.1

Sat Jan 17 2015 14:22:55

Contents

Chapter 1

Class Index

1.1 Class List

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

CameraSystem . . . . ??
Entity . . . . ??
GameManager . . . . ??
Matrix3 . . . . ??
MovementSystem . . . . ??
PlayerInputSystem . . . . ??
RenderSystem . . . . ??
ResourceManager . . . . ??
Scene . . . . ??
ShaderData . . . . ??
ShaderInterface . . . . ??
ShaderLoader . . . . ??
Vector2 . . . . ??
Vector3 . . . . ??
Vector4 . . . . ??
VertexBuffer . . . . ??
VertexDataP . . . . ??
VertexDataPN . . . . ??

Chapter 2

Class Documentation

2.1 CameraSystem Class Reference

Public Member Functions

- Entity ∗ **getCurrentCamera** ()
- void setCurrentCamera (Entity ∗newCamera)

  Create Camera entity.

Static Public Member Functions

- static CameraSystem & **getCameraSystem** ()
- static void destroyCameraSystem ()

  Destruct Camera entity.

2.1.1 Member Function Documentation

2.1.1.1 void CameraSystem::destroyCameraSystem ( ) [static]

Destruct Camera entity.

Gets the camera and deletes it.

2.1.1.2 void CameraSystem::setCurrentCamera (Entity ∗ *newCamera*)

Create Camera entity.

Creates the camera entity to be added to the scene

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

- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/CameraSystem.h
- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/CameraSystem.cpp

2.2 Entity Class Reference

Public Member Functions

- VertexBuffer ∗ **getVertexBuffer** ()

- void **setVertexBuffer** (VertexBuffer ∗newVertexBuffer)
- Vector3 **getPosition** ()
- void **setPosition** (Vector3 newPosition)
- Vector3 **getScale** ()
- void **setScale** (Vector3 newScale)
- Vector3 **getRotation** ()
- void **setRotation** (Vector3 newRotation)
- Vector3 **getVelocity** ()
- void **setVelocity** (Vector3 newVelocity)
- Vector3 **getScaleVelocity** ()
- void **setScaleVelocity** (Vector3 newScaleVelocity)
- Vector3 **getRotationVelocity** ()
- void **setRotationVelocity** (Vector3 newRotationVelocity)
- Vector3 **getEyeVector** ()
- void **setEyeVector** (Vector3 newEyeVector)
- Vector3 **getUpVector** ()
- void **setUpVector** (Vector3 newUpVector)
- Entity (VertexBuffer ∗vertexBuffer, Vector3 position)

    Entity.

2.2.1 Constructor & Destructor Documentation

2.2.1.1 Entity::Entity (**VertexBuffer** ∗ *vertexBuffer,* **Vector3** *position*)

Entity.

Performs matrix transormation on all scene elements.

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

- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/Entity.h
- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/Entity.cpp

2.3 GameManager Class Reference

Public Member Functions

- void **runGameLoop** ()

Static Public Member Functions

- static GameManager & getGameManager ()

    GameManager create.
- static void destroyGameManager ()

2.3.1 Member Function Documentation

2.3.1.1 void GameManager::destroyGameManager ( ) [static]

Gets the current game and deletes it.

Console output during development, not needed - May remove.

Gets the context window and destroys

End glfw and free memory.

2.3.1.2 GameManager & GameManager::getGameManager ( ) [static]

GameManager create.

Creates all basic window context and colour settings. Create instance of game.

Ensures that the following code will not run if gameManager has been instantiated.

Initialise window.

Set red bit depth to a value of 8.

Set green bit depth to a value of 8.

Set blue bit depth to a value of 8.

Set alpha bit depth to a value of 8.

Window is set to a fixed size.

A pure virtual member.

See also

glfwCreateWindow

Parameters

|>p0.15|p0.805|

1280	Screen Width.
720	The screen height.
NULL,NULL	- Not used, as options for multiple displays.

Screen settings glfwCreateWindow(width, height, title...

The other two parameters are used for multiple display setups and will not be used.

Creates the context window, the game window is now visible.

Console output - Not needed, may remove.

Return the window/context to be set to run in main.

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

- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/GameManager.h
- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/GameManager.cpp

2.4 Matrix3 Struct Reference

Public Attributes

- GLfloat **m00**
- GLfloat **m01**
- GLfloat **m02**
- GLfloat **m10**
- GLfloat **m11**
- GLfloat **m12**
- GLfloat **m20**
- GLfloat **m21**
- GLfloat **m22**

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

- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/Matrix3.h

2.5 MovementSystem Class Reference

Public Member Functions

- void update (std::vector$<$ Entity $*>$ $*$entityArray)

  Movement class.

Static Public Member Functions

- static MovementSystem & getMovementSystem ()

  getMovementSystem
- static void destroyMovementSystem ()

  destroyMovementSystem

2.5.1 Member Function Documentation

2.5.1.1 void MovementSystem::destroyMovementSystem ( ) `[static]`

destroyMovementSystem

Gets the movement system and then deletes it and frees up memory.

2.5.1.2 **MovementSystem** & MovementSystem::getMovementSystem ( ) `[static]`

getMovementSystem

If there is no movement system, one will be created and returned.

2.5.1.3 void MovementSystem::update (std::vector$<$ **Entity** $*>$ $*$ *entityArray*)

Movement class.

Uses the transforms returned by the entity to perform player movement.

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

- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/MovementSystem.h
- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/MovementSystem.cpp

2.6 PlayerInputSystem Class Reference

Public Member Functions

- void **setCurrentPlayer** (Entity $*$newPlayer)
- void update ()

  Keyboard controls.

Static Public Member Functions

- static PlayerInputSystem & **getPlayerInputSystem** ()
- static void **destroyPlayerInputSystem** ()
- static void **keyCallbackFun** (GLFWwindow ∗window, int key, int scancode, int action, int mods)

2.6.1 Member Function Documentation

2.6.1.1 void PlayerInputSystem::update ( )

Keyboard controls.

Parameters

|>p0.15|p0.805|

W	increase Z position(move forward)
S	decrease z position(move backwards)
A	Perform matrix transform and subtract from position (Move left)
D	Perform matrix transform and add to position (Move right)

Mouse controls

Uses mouse position cooridinates on screen and matrix transformations to calculate the angle needed after mouse movement to adjust the camera position.

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

- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/PlayerInputSystem.h
- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/PlayerInputSystem.cpp

2.7 RenderSystem Class Reference

Public Member Functions

- Entity ∗ getCurrentCamera ()

  Grabs current camera entity.
- void **setCurrentCamera** (Entity ∗newCamera)
- void render (std::vector< Entity ∗ > ∗entityArray)

  Render Scene.

Static Public Member Functions

- static RenderSystem & getRenderSystem ()
- static void destroyRenderSystem ()

  Render System delete.

2.7.1 Member Function Documentation

2.7.1.1 void RenderSystem::destroyRenderSystem ( ) [static]

Render System delete.

Deletes the render system.

2.7.1.2 Entity ∗ RenderSystem::getCurrentCamera ( )

Grabs current camera entity.

Returns camera variable.

2.7.1.3 RenderSystem & RenderSystem::getRenderSystem ( ) [static]

gluPerspective

Parameters

|>p0.15|p0.805|

Field of view measured in degrees.

Aspect ratio, simpl the width/height.

Near clipping point.

Far clipping point i.e. draw distance.

View port setup

Parameters

|>p0.15|p0.805|

X and Y for bottom left corner of the viewport (0,0).

Width of the viewport.

Height of the Viewport.

Cull faces

Removes faces not visible within the viewport.

2.7.1.4 void RenderSystem::render (std::vector< Entity ∗ > ∗ *entityArray*)

Render Scene.

Takes all position data from the camera and cube entities and applies the model, view projections matrices to generate the scene. Entity iterator

Iterates through the Entity array to render each entity object.

Swap buffers

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

- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/RenderSystem.h
- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/RenderSystem.cpp

2.8 ResourceManager Class Reference

Public Member Functions

- std::vector< ShaderInterface ∗ > ∗ getShaderArray ()

  Get Shader Array.
- std::vector< VertexBuffer ∗ > ∗ getVertexBufferArray ()

Get Vertex Array.

Static Public Member Functions

- static ResourceManager & getResourceManager ()

  Get resource manager method.
- static void destroyResourceManager ()

  destroy Resource Manager

2.8.1 Member Function Documentation

2.8.1.1 void ResourceManager::destroyResourceManager ( ) [static]

destroy Resource Manager

Deletes the resource manager and frees up the memory.

2.8.1.2 ResourceManager & ResourceManager::getResourceManager ( ) [static]

Get resource manager method.

Returns the resource manager.

2.8.1.3 std::vector< ShaderInterface ∗ > ∗ ResourceManager::getShaderArray ( )

Get Shader Array.

Returns the variable _shader array.

2.8.1.4 std::vector< VertexBuffer ∗ > ∗ ResourceManager::getVertexBufferArray ( )

Get Vertex Array.

Returns the variable _vertexBufferArray

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

- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/ResourceManager.h
- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/ResourceManager.cpp

2.9 Scene Class Reference

Public Member Functions

- std::vector< Entity ∗ > ∗ **getChildren** ()
- Scene ()

  Scene Creator.
- ∼Scene ()

  Scene Destructor.

2.9.1 Constructor & Destructor Documentation

2.9.1.1 Scene::Scene ( )

Scene Creator.

Creates and populates the scene. Create instance of resource manager

Creates a resource manager to deal with generation of the cubes

Create a cube Entity

Takes the paramaters :-

Parameters

|>p0.15|p0.805|

X-Pos	a float value for the x-position in the scene.
Y-Pos	a float value for the Y-position in the scene.
Z-Pos	a float value for the Z-position in the scene.

Create a camera for the scene

Generates a camera for the scene at world space origin.

Setup Player input

Applys player input system controls, onto the camera.

2.9.1.2 Scene::∼Scene ( )

Scene Destructor.

Destructs the scene by deleting each item/child created, freeing them from memory.

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

- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/Scene.h
- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/Scene.cpp

2.10 ShaderData Class Reference

Public Member Functions

- Vector4 **get_uColorValue** ()
- void **set_uColorValue** (Vector4 newColor)
- Vector3 **get_uLightPosition** ()
- void **set_uLightPosition** (Vector3 newPosition)
- **ShaderData** (Vector4 newColor, Vector3 newPosition)

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

- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/ShaderData.h
- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/ShaderData.cpp

2.11 ShaderInterface Class Reference

Public Member Functions

- GLuint **getProgramHandle** ()
- GLint **get_aPositionVertex** ()
- GLint **get_aPositionNormal** ()
- GLint **get_uColor** ()
- GLint **get_uLightPosition** ()
- ShaderInterface (const char ∗VS, const char ∗FS)

  Shader Interface.
- ∼ShaderInterface ()

  Shader Interface Deconstructor.

2.11.1 Constructor & Destructor Documentation

2.11.1.1 ShaderInterface::ShaderInterface (const char ∗ *VS*, const char ∗ *FS*)

Shader Interface.

Takes the shader files applies them to the Shader loader. new ShaderLoader

loads the vertex and fragment shader text files into the shader loader.

Free shader

Release the shader text files from memory.

2.11.1.2 ShaderInterface::∼ShaderInterface ( )

Shader Interface Deconstructor.

Delets the shader.

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

- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/ShaderInterface.h
- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/ShaderInterface.cpp

2.12 ShaderLoader Class Reference

Public Member Functions

- GLuint getProgramHandle ()

  getProgramHandle
- ShaderLoader (const char ∗sourceVS, const char ∗sourceFS)

  Shader loader.
- ∼ShaderLoader ()

  Shader Loader Deconstructor.

2.12.1 Constructor & Destructor Documentation

2.12.1.1 ShaderLoader::ShaderLoader (const char ∗ *sourceVS*, const char ∗ *sourceFS*)

Shader loader.

Loads in the vertex and fragment shader for use in the game engine. Vertex Shader

Compiles the source vertex shader into the openGL vertex shader.

Fragment Shader

Compiles the source vertex shader into the openGL fragment shader.

glAttachShader

Attaches the shader to the program handle.

glLinkProgram

Links the program handle using openGL link program.

Delete the shaders

Now the shaders have been compiled and linked to the program, can now free up the memory used.

2.12.1.2 ShaderLoader::∼ShaderLoader ( )

Shader Loader Deconstructor.

Deletes the program and frees up the memory.

2.12.2 Member Function Documentation

2.12.2.1 GLuint ShaderLoader::getProgramHandle ( )

getProgramHandle

Returns the program handle

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

- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/ShaderLoader.h
- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/ShaderLoader.cpp

2.13 Vector2 Struct Reference

Public Attributes

- GLdouble **x**
- GLdouble **y**

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

- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/Vector2.h

2.14 Vector3 Struct Reference

Public Attributes

- GLfloat **x**
- GLfloat **y**
- GLfloat **z**

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

- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/Vector3.h

2.15 Vector4 Struct Reference

Public Attributes

- GLfloat **x**
- GLfloat **y**
- GLfloat **z**
- GLfloat **w**

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

- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/Vector4.h

2.16 VertexBuffer Class Reference

Public Member Functions

- GLuint **getVertexBufferID** ()
- ShaderInterface ∗ **getShader** ()
- ShaderData ∗ **getShaderData** ()
- **VertexBuffer** (const GLvoid ∗data, GLsizeiptr size, GLenum mode, GLsizei count, GLsizei stride, Shader←Interface ∗shader, ShaderData ∗shaderData, GLvoid ∗positionOffset, GLvoid ∗normalOffset)
- ∼VertexBuffer ()

  Vertex Buffer destructor.
- void configureVertexAttributes ()

  configure Vertex Attributes
- void renderVertexBuffer ()

  render Vertex Buffer

2.16.1 Constructor & Destructor Documentation

2.16.1.1 VertexBuffer::∼VertexBuffer ( )

Vertex Buffer destructor.

Deletes all vertex buffers and sets the vertexBuffer variable to 0.

2.16.2 Member Function Documentation

2.16.2.1 void VertexBuffer::configureVertexAttributes ( )

configure Vertex Attributes

When _shader is empty (-1 position) grab a new position vertex and apply this to the vertex attributes.

2.16.2.2 void VertexBuffer::renderVertexBuffer ( )

render Vertex Buffer

Calls for the open gl to draw arrays

Parameters

|>p0.15|p0.805|

Primitve	render type
Start	position (0 start)
Size	of array vertices.

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

- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/VertexBuffer.h
- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/VertexBuffer.cpp

2.17 VertexDataP Struct Reference

Public Attributes

- Vector3 **positionCoordinates**

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

- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/VertexData.h

2.18 VertexDataPN Struct Reference

Public Attributes

- Vector3 **positionCoordinates**
- Vector3 **normalCoordinates**

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

- /Users/mada360/Desktop/OpenGLCube/OpenGLCube/VertexData.h