GAME ENGINE PROGRAMMING ASSIGNMENT 2

Implementation Report

Richard Hancock
April 2017

Table of Contents

Richard Hancock – i7249671

Introduction

For this Assignment, I have focused on developing new and expanding upon existing features in my Game Engine. I primarily focused on Cross Platform, Networking and Physics.

UML Diagrams

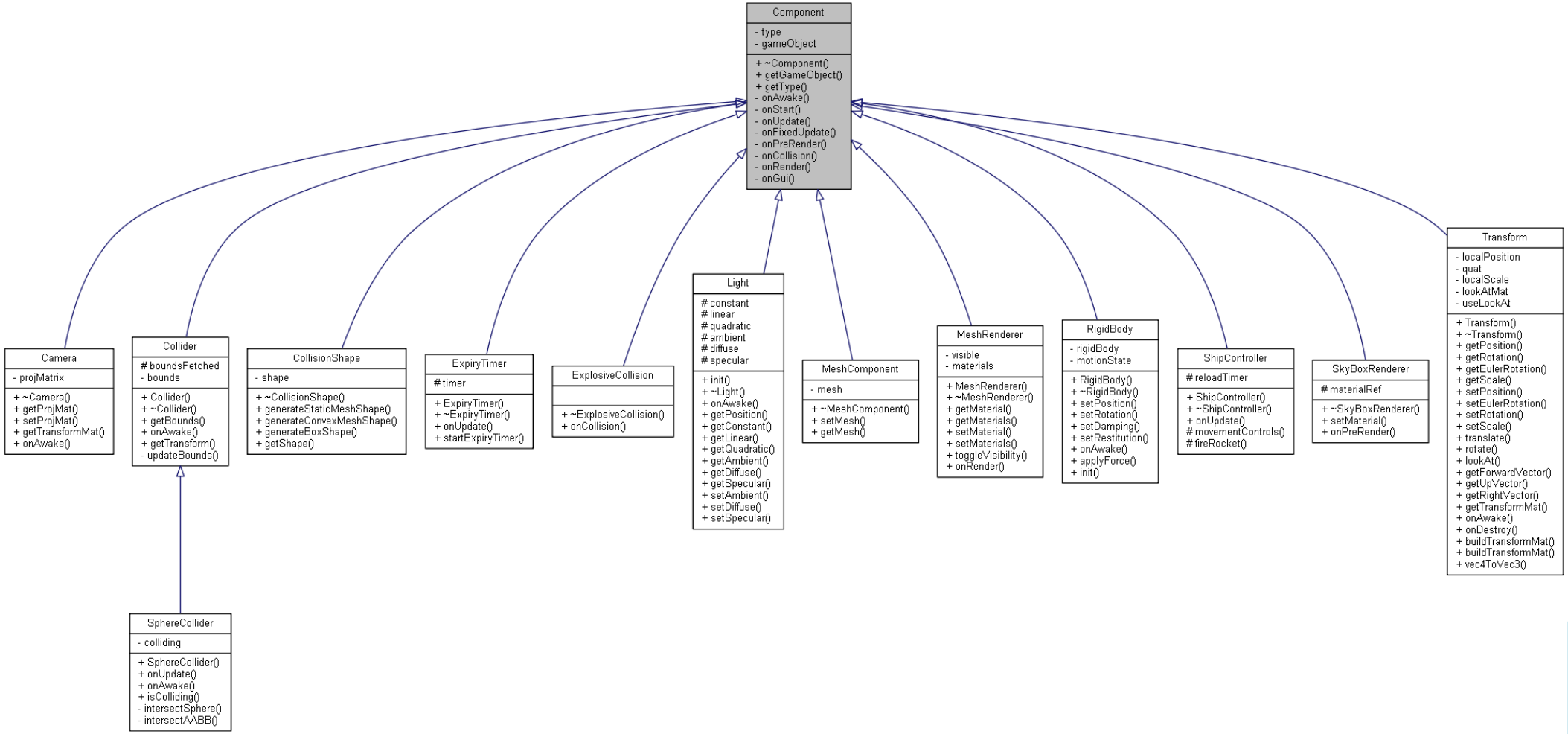

Figure 1 UML Diagram of all Component classes in my Engine currently. (Note: Collider and Sphere Collider are my own physics and are not used anymore)

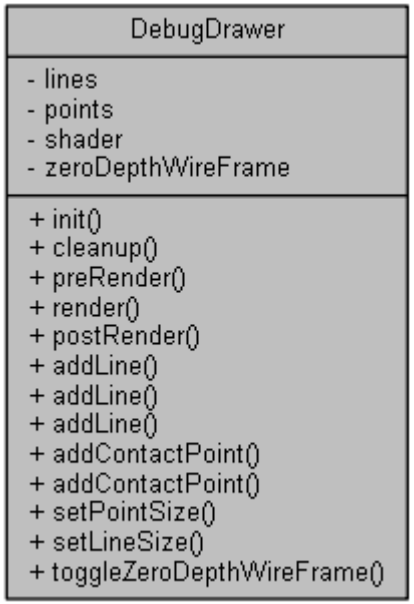

Figure 2 UML Diagram of the Debug Drawer which can be used to draw simple primitives such as lines and points. Bullet can utilize this to display all the physics data it has available.

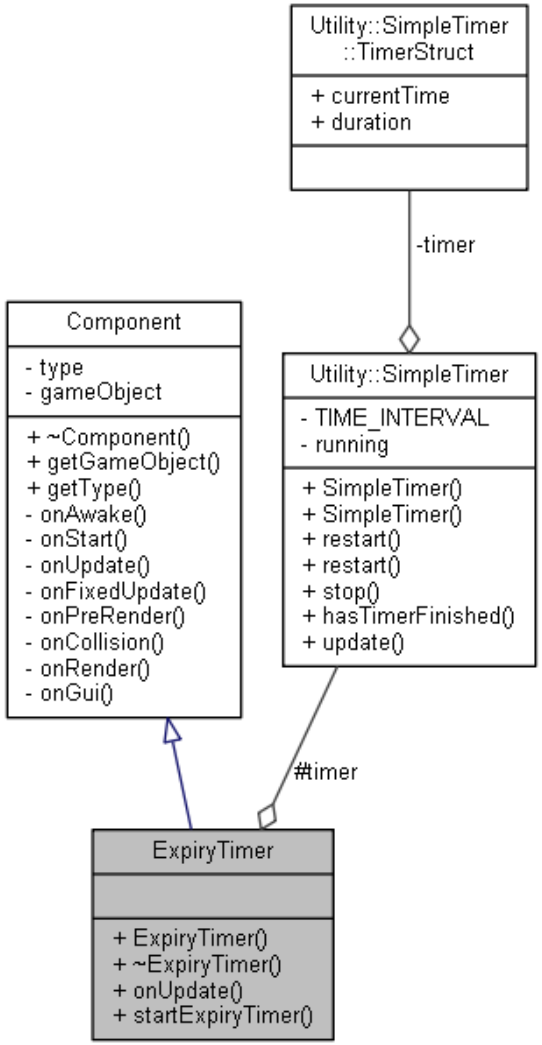

Figure 3 UML Diagram showing how the ExpiryTimer is made up. The expiry time is a simple component that contains a timer, once that time has finished it will delete the game object it is attached to. (I use this for the ships rockets).

Features

- **Cross Platform on Windows, Linux and Mac** with a custom python scripted build system that allows for the easy generation of Makefiles and even fetching of dependencies in some cases. Adding Mac support was difficult as I don't have any experience with or own a Mac, so I had to rent a Mac server, teach myself the operating system and then port the code over to it. Unfortunately, the Mac server I use is also very slow (~30 mins to build the code) so I can only barely run the game engine on it at all.

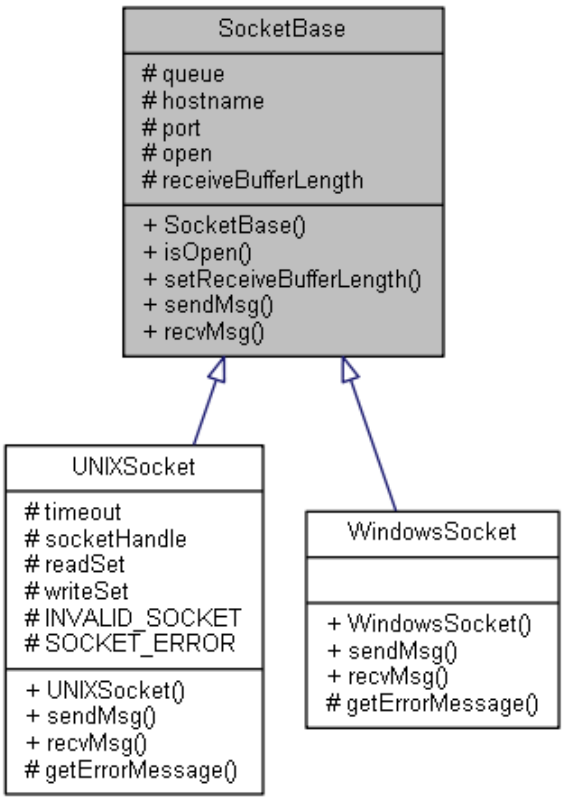

Figure 4 UML Inheritance diagram showing how the sockets derive from a common base class

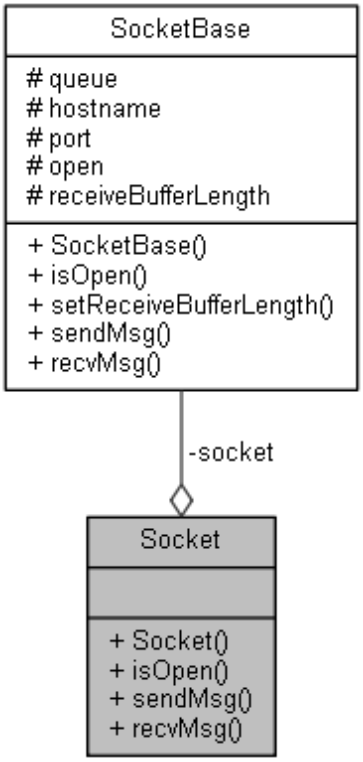

Figure 5 UML Collaboration diagram showing the Socket interface which is the only part an end user will interact with.

- **Cross Platform networking,** I have implemented both WinSock and BSD sockets in the engine and can be accessed by a standardized interface, with the engine doing all the platform specific networking behind the scenes. However, I do not currently have a game server component and have been testing the networking with IRC and web servers. I almost had a system that uploaded user data (logs, platform specs, general analytics) to a web page, but had to disable it as it needs more work to be stable.

Figure 6 Height Map

- **Height Map Loader** that can convert a grayscale image into geometry and height data with the option of texturing. Can scale the map in height or dimensions to increase land size (without requiring increased height map image resolution).
- **Bullet Physics Implementation,** I have implemented most of the core functionality of Bullet:
    - **Triangular Mesh Collision:** My Game Model class can be converted straight into a triangular mesh for bullet to use in collision.
    - **Collision Callbacks:** When a collision occurs, I trigger an onCollision event to the affected Game Objects and their components.
    - **Quaternion Rotations:** Due to recurring problems with Euler angles I implemented Bullet's quaternion class to handle rotations for all objects. (I originally tried to use GLM but its data did not match bullet's, thus causing weird behaviour).

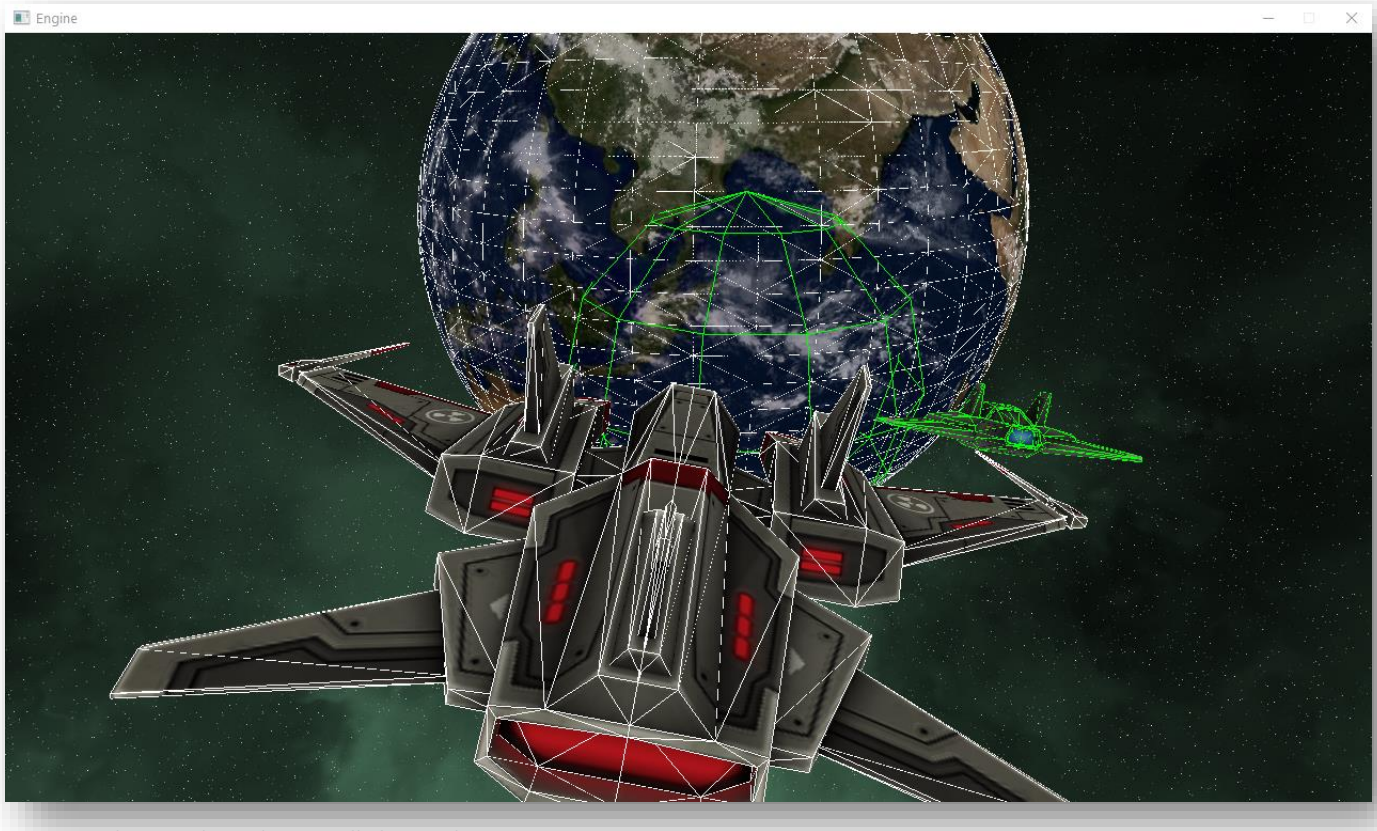

Figure 7 Debug Renderer showing all physics objects

- **Debug Drawer/Renderer:** Bullet provides an interface that if implemented will allow all physics objects to be displayed on screen as a wireframe, showing how each collision shape wraps its assigned model. I built this debug drawer to use modern OpenGL techniques (Rather than 'glDrawLine') and to be independent of Bullet so it could be used to render any primitive data.

Figure 8 Portion of the SkyBox

- **SkyBoxes and Cube maps,** Cube maps have been added on the OpenGL side to allow for 6 textures that make up the faces of a cube. I have mainly implemented these to add a SkyBox or StarScape Renderer to replace the single plane I used to display the background previously. Each of the 6 textures is 4096x4096 so displays a lot of detail.
- **WebSocket support (Dissertation),** Implemented into the engine as part of my dissertation allows mobile devices to be used as secondary displays and controllers. While the code is still in the submission I have disabled the feature as it cannot run without a web server component that I can only host as required.

User Guide

Settings can be changed in the settings file which is stored at:
"C://Users/NAME/AppData/Roaming/RH/Engine"

Once the program has loaded, the ship can be controlled using:

- W – Move forward
- S – Move backwards
- A – Move left
- D – Move right
- Q – Move down
- E – Move right
- Arrow keys – Rotate
- K, L – Roll

Alternatively, the Left analog stick of a connected controller can move the camera in some axis.

Other Controls

- F – Enter planet (Currently only one way, will need to reload game to go back to space).
- B – Drop Bouncy Balls (Only works on planet)
- Spacebar – Fire Rocket (These cause explosions that can propel other objects out of its blast radius (visible using debug view).
- F1 – Toggle Debug View (Shows all of bullets physics objects) WARNING: Can slow performance to a crawl during heavy physics.

Evaluation

The code compiles with no warnings (Apart from 3rd party libraries) on level 4 in Visual Studio. It also compiles with almost no warnings on G++(GCC) and Clang on Ubuntu and Mac OS Sierra. The engine runs at a reasonable performance on Windows and Linux, however I cannot test Mac's performance due to the problems I mentioned above (but it does run).

Conclusions

In conclusion, I would have liked to get more done during this assignment as Bullet delayed me significantly with its lack of documentation and maintenance. In the future, I would not use Bullet again, preferring NVidia's PhysX or my own implementation instead.
I would have also liked to be able to test my engine on Mac more and possibly port it to Emscripten and Android.

In summary, I am pleased that Bullet's physics work but I will be looking for alternative physics libraries in the future that allow easier integration and are actively maintained/documented.

Appendices

In addition to this report I have generated full DoxyGen documentation in both HTML and PDF formats. I have turned all the UML settings to their highest to provide inheritance, collaboration, usage and file diagrams.