# MMG Engine

S5063022 - Oscar Lennox-Hilton

## Overview

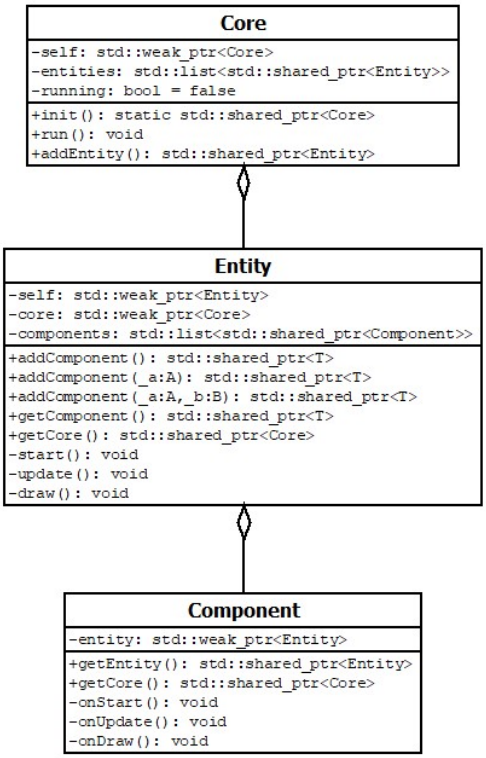
MMG (MeMakeGame) is an entity-component-based game engine. The engine renders graphics to a game window and supports mouse and keyboard input. It provides exception handling throughout its pipelines to allow the user to more easily debug the game they are building.

Many design decisions are influenced by the Unity engine, being derived from the same Entity Component System. However, whereas Unity is a closed-source game engine software with C# scripting, MMG is an open-source C++ library.

MMG utilises the following dependencies:

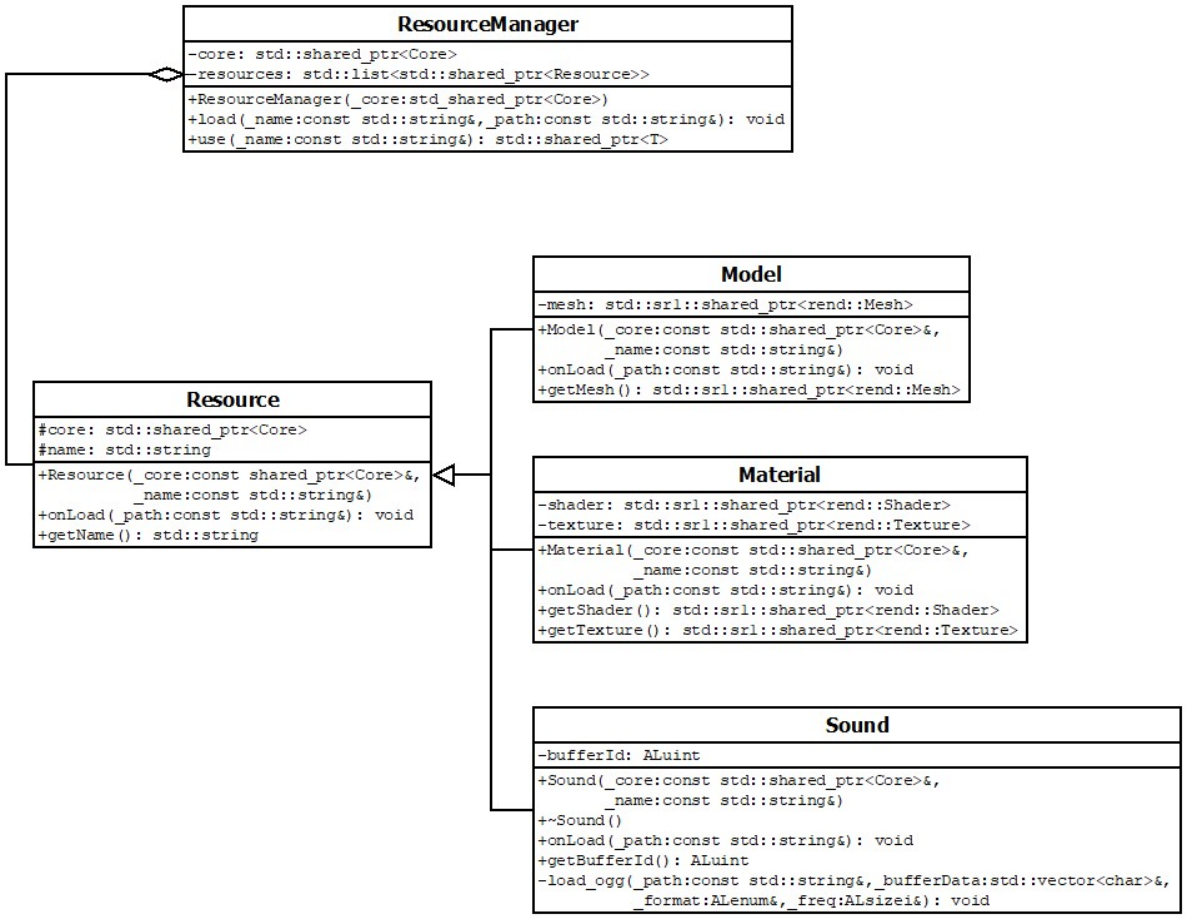
- SDL2 for window and input management
- OpenGL for 3D graphics rendering
- GLEW for OpenGL extension loading support
- GLM for maths functions
- OpenAL for 3D audio
- stb_image for image file loading
- stb_vorbis for audio file loading

## Entity Component System




The Entity Component System that MMG follows is based on the principle that all objects in the game are Entities, contained within a Core. Entities by themselves have no gameplay behaviour but are instead flexible vessels for Components to add behaviours to. This way, every object in the game can be looped through in Core to call all update(), draw(), etc. methods in a single game loop.

## Resource system

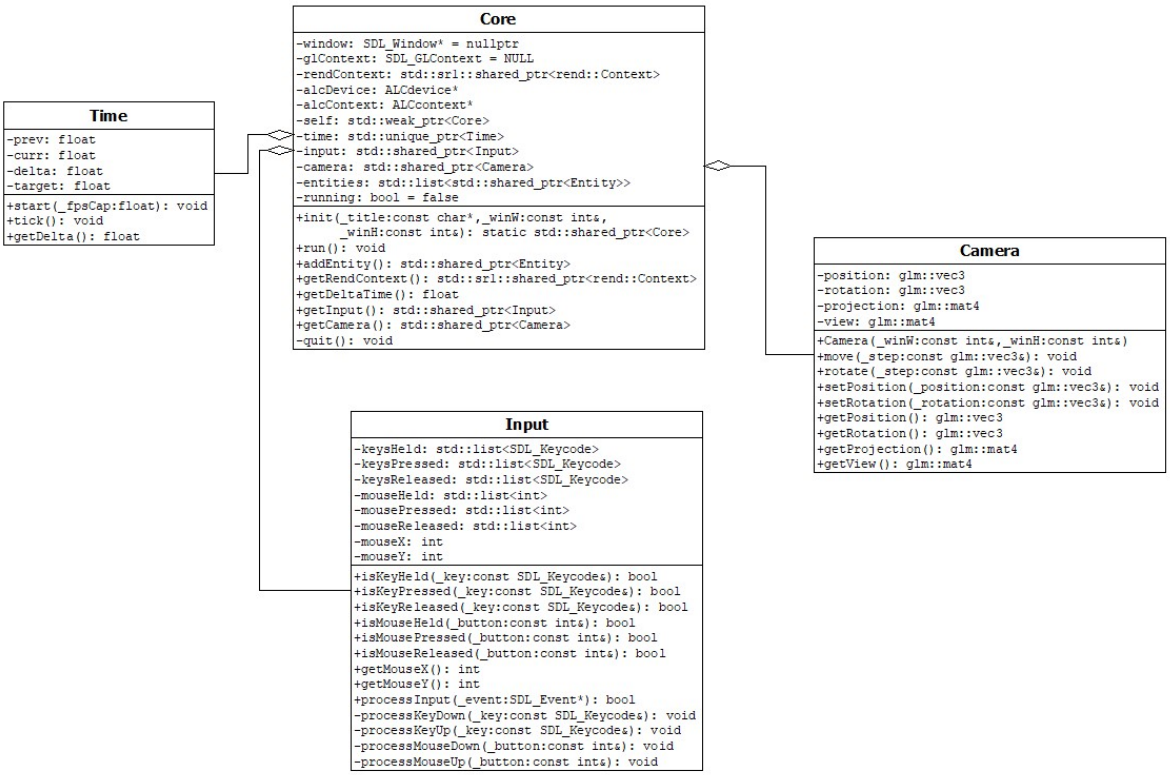



Assets are loaded through MMG into the game through the ResourceManager.

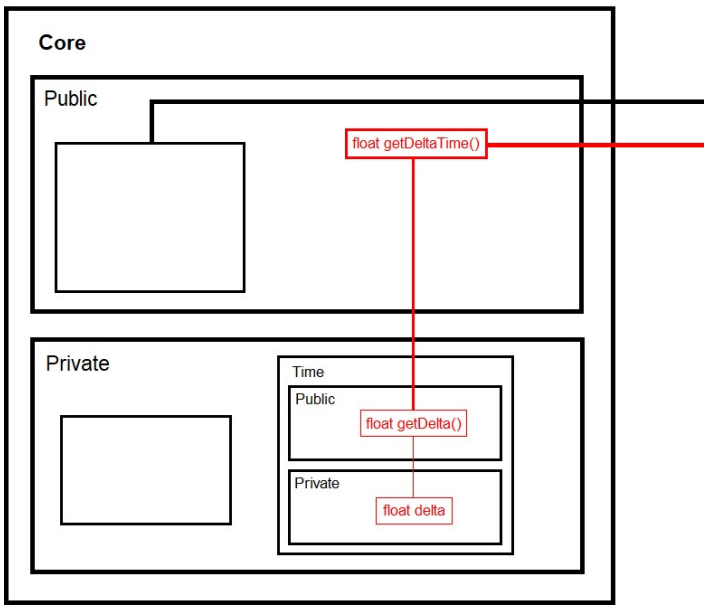
The ResourceManager contains a list of all Resources loaded from files, where Resource is the base class, and each type of asset is derived from it. Resources may be accessed from other parts of the game engine through the ResourceManager.

When loading a resource, if the file cannot be found at the specified path, the engine will handle the exception and print the error to the console.

## Core




The Core class also contains references to other parts of the game engine; the camera, the input handler, and a Time class. The public interfaces of these classes can be accessed through getters in Core, as depicted below.



## Conclusion

The engine's Entity Component System is robust and extendable, and mouse and keyboard input are handled efficiently and intuitively. However, there is a lot of pointer chasing involved when making a game with MMG, and simple commands quickly become long-winded and tedious to type. This could be improved by taking a different approach to how each piece of the engine is encapsulated, instead of most pieces being contained in Core (which itself must be accessed through getEntity when inside a Component).

Currently, the feature set provided by MMG is very minimal. There is a lot of room for additional features, such as UI components, mesh collision, and menu management.