**Game Engine Req Systems**

Table of Contents

**Entry Point2**

Type chapter title (level 2)2

Type chapter title (level 3)3

**Application Layer4**

**Window Layout4**

Input and Events5

**Renderer4**

**Render API Abstraction4**

**Debugging Support4**

**Scripting Language4**

**Memory Systems4**

**Entity-Component System (ECS)4**

**Physics4**

**File I/O & Virtual File System (VFS)4**

**Build System4**

**Setup:**

We will need to build engine solution in to a Dynamic Link Library, or .dll

And create an external application/ .exe and link the .dll engine file to it.

The .dll will contain all the necessary dependencies to run the game.

In VS, make c++ project and set output file to be .dll in project properties.

Then add a new c++ project to the current solution. This will be the game project. Named Sandbox.

Go into .sln file for engine and move it above the .dll engine such that the .exe runs first.

Right click on sandbox and add reference to .dll engine. Now, when .dll engine compiles and generates static library file + .dll file, it will link the .exe solution with it

**ENTRY POINT**

Entry Point is the behavior that occurs when the game engine launches.

APPLICATION LAYER

Application layer is the section that handles application life cycle, events, window resizing, controller input, etc. such as the run() loop, which allows the application to open and render frames.

**WINDOW LAYOUT**

Events and Input

**RENDERER**

The largest part of the engine.

**RENDERER API AND ABSTRACTION**

The goal is to support a variety of graphics APIs by remaining API agnostic.

**DEBUGGING SUPPORT**

Structure to easily allow for logging errors, values, and performance

**SCRIPTING LANGUAGE**

Allow for scripting in other languages.

**MEMORY SYSTEMS**

Inspect, minimize, and manage memory usage.

**ENTITY-COMPONENT SYSTEM (ECS)**

Allow the game programmer to modularize entities by defining behavior with components.

**PHYSICS**

Mathematically emulating real world or programmer intended physics.

**FILE I/O & VIRTUAL FILE SYSTEM (VFS)**

File system to store local data and such.

**BUILD SYSTEM**

Build method to convert file types to required type outside of system, such that it doesn’t waste resources converting every time the application runs.