Systems & System Manager

The whole architecture of the engine relies on systems. A system is a **static** class with a single instance held in a System Manager. A system refers to a single section of data such as Events or Windows. All systems are added during the engine’s construction and is the last object to be deleted in the engine.

Systems are stored in a **vector**, so the order of insertion is remembered and important. Systems are destroyed in reverse order of insertion.

Current systems:

* Log
* Randomiser
* Timer System
* Window API System
* Window Manager
* Event Manager
* Resource Manager
* Scene Manager
* Font Manager
* Render System

When a system is added to the system manager, the system’s **start** function is called. When the systems are destroyed, their **stop** functions are called.

**System Manager**

The system manager must be **initialised** and requires the **maximum** number of systems to support. Systems can be added to the system manager through the **addSystem** function. Only **one** instance of a type of system can be added.

**Adding New Systems Instructions**

1. The new system type must be declared in the system type enum in system.h
2. Add the system include to systemManager.h
3. Update the switch case in the addSystem function in systemManager.cpp
4. Add the system in the Application constructor (Update system manager max count if limit has been reached)