Engine API UML

SceneSubSystem

<<Interface>>
IManager
+Initialize() : void
+End() : void
+Update(DeltaTime : float) : void

Scene
+EntityComponentManager : std::unique_ptr<SECManager>
+BeginScene() : void
+EndScene() : void
+Update(DeltaTime : float)
+SetupEngineEvents(Events : EventAttorney&) : void

SceneManager
-Scenes : std::unordered_map<std::string, std::unique_ptr<Scene>>
+Initialize() : void
+End() : void
+Update(DeltaTime : float) : void
+AddScene(Name : std::string) : Scene*
+GetScene(Name : std::string) : Scene*
+RemoveScene(Name : std::string) : Scene*
+SetupEngineEvents(Events : EventAttorney&) : void

<<Interface>>
IManager
+Initialize() : void
+End() : void
+Update(DeltaTime : float) : void

EventAttorney
+Initialize() : void
+OnInput : Signal<InputEventType>*
+OnKeyDown : Signal<KeyCode>*
+OnKeyHold : Signal<KeyCode>*
+OnKeyUp : Signal<KeyCode>*

EventSubSystem

<<Interface>>
IManager
+Initialize() : void
+End() : void
+Update(DeltaTime : float) : void

Signal   `Args : ...T`
-Slots : std::unordered_map<unsigned int, std::...
+Connect(Slot : std::function<...>) : unsigned int
+Disconnect(SlotID : unsigned int) : void
+EmitAll(Arguments : Args...) : void

Engine
-Running : bool
-EngineSceneManager : SceneManager
-EngineWindowManager : WindowManager
-EngineEventManager : EventManager
-EngineRenderManager : RenderManager
-WindowInfo : WindowInfo
+EndEngine() : void
+StartEngine() : void
+Update(DeltaTime : float) : void
+SetWindowInfo(WindowInfo : WindowInfo&&) : void

<<instantiate>>

Game entry point

<<Abstract>>
EventManager
+OnInput : Signal<InputEventType>
+OnKeyDown : Signal<KeyCode>
+OnKeyHold : Signal<KeyCode>
+OnKeyUp : Signal<KeyCode>
+Update(DeltaTime : float) : void

<<enumeration>>
InputEventType
DOWN
UP
HOLD

<<enumeration>>
KeyCode
KEY_A
KEY_B

SDLEventManager
+Initialize() : void
+End() : void

WindowSubSystem

WindowInfo
+Width : size_t
+Height : size_t
+PosX : size_t
+PosY : size_t
+Title : std::string
+WindowInfo(Width : size_t, Height : size_t, PosX : size_t, PosY size_t, Title : std::string&&)
+Prototype() : WindowInfo

<<Interface>>
IManager
+Initialize() : void
+End() : void
+Update(DeltaTime : float) : void

<<Interface>>
IWindow
+CreateWindow(WindowInfo : WindowInfo&&) : void
+DestroyWindow() : void
+SetAsContext() : void
+Update(DeltaTime : float) : void

WindowManager
-window : std::unique_ptr<IWindow>
+WindowManager(window : std::unique_ptr<IWindow>&&)
+Initialize() : void
+End() : void
+Update(DeltaTime : float) : void
+CreateWindow(WindowInfo : WindowInfo&&) : void

GLSDLWindow
+CreateWindow(WindowInfo : WindowInfo&&) : void
+DestroyWindow() : void
+SetAsContext() : void
+Update(DeltaTime : float) : void

use

ManagerFactory
+CreateSceneManager() : std::unique_ptr<SceneManager>
+CreateWindowManager() : std::unique_ptr<WindowManager>
+CreateEventManager() : std::unique_ptr<EventManager>
+CreateRenderManager() : std::unique_ptr<RenderManager>

RenderSubSystem

<<Interface>>
IManager
+Initialize() : void
+End() : void
+Update(DeltaTime : float) : void

<<Abstract>>
RenderManager
+CreateRenderer() : Renderer

<<Interface>>
IRenderer

GLSDLRenderManager
+CreateRenderer() : Renderer
+Initialize() : void
+End() : void
+Update(DeltaTime : float) : void