1. lua实现框架是结合开源mvc框架puremvc来进行简化实现，抽取了其中的Controller、Mediator(中介者模式)和Proxy(代理模式)的概念，用于管理控制器、视图和模型
2. 关于框架中WWFacade的说明

此类也是取自puremvc框架的概念，用于处理框架的核心交互与控制 – 事件监听与派发(观察者模式的运用)

框架中所有的事件的监听与派发最终都是此类来完成，此类中包含了一个lua层框架级的EventDispatcher实例，与cocos引擎自身Director提供的EventDispatcher是独立的，互不影响的，类中也提供了Global相关的接口来处理Director提供的EventDispatcher的相关操作

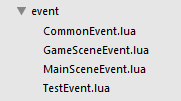
1. 关于框架中管理类的说明：



ControllerRegistry、MediatorRegistry和ProxyRegistry用于注册框架启动时，需要实例化的Controller、Mediator和Proxy子类

ControllerManager、MediatorManager和ProxyManager用于初始化所有注册的Controller、Mediator和Proxy子类实例并提供对应的访问查询接口，用于获取对应的参数指定的Controller、Mediator和Proxy子类实例

1. 游戏逻辑中使用的事件全部定义在event目录中，不同模块的事件定义在对应的模块名组成的文件中



1. Controller、Mediator和Proxy的关系与职责

Controller用于处理Mediator和Proxy之间的交互与对应接口调用

提供监听事件、注销事件听监听以及派发事件功能

提供查询和访问所有已注册的Mediator和Proxy实例的功能，也提供了获取其它Controller实例的功能(此功用用处不大)

主要用于新模块启动的时候Mediator(界面)和Proxy(数据-模型)分别初始化，

Mediator用于管理此模块下不同的View之间的逻辑交互，以及与Proxy交互

提供了监听事件，注销事件监听以及派发事件功能

提供了查询和访问所有已注册的Proxy实例的功能，用于获取不同模块之间的数据，来进行游戏逻辑的实现

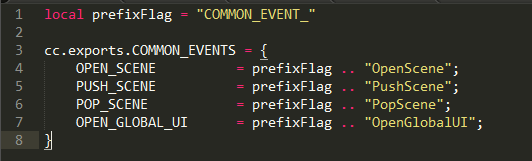
Proxy用于管理此模块下游戏数据的缓存以及与服务端的通信交互

提供对应的数据设置与获取接口

提供事件派发功能(此模块不需要监听事件，只需要对应的数据刷新或是修改的时候派发对应的事件通知Mediator或是Controller)

另外框架中添加了全局的SceneMediator用于管理场景的切换与管理

对应的事件定义在CommonEvent.lua文件中



主要实现了C++端WWSceneManager相关的场景的管理功能

实例说明：(此实例实现了之前游戏demo中的游戏逻辑)

实例由两个界面组成主界面和游戏界面

主界面相关文件列表：

app.controller.MainScene.MainSceneController.lua -- Controller子类

app.mediator.MainScene.MainSceneMediator.lua -- Mediator子类

app.proxy.MainScene.MainSceneProxy.lua -- Proxy子类

app.event.MainSceneEvent.lua -- 模块事件定义

游戏界面相关文件列表

app.controller.GameScene.GameSceneController.lua -- Controller子类

app.mediator.GameScene.GameSceneMediator.lua -- Mediator子类

app.proxy.GameScene.GameSceneProxy.lua -- Proxy子类

app.event.GameSceneEvent.lua -- 模块事件定义

主界面模块定义了两个事件，一个是进入此模块的入口事件Entry和一个模拟登陆相关的Login事件，在MainSceneController中初始化时会注册两个事件的监听，

游戏界面模块定义了两个事件，一个是进入此模块的入口事件Entry和一个登陆成功的事件，在GameSceneController中初始化时会注册Entry事件的监听，在GameSceneMediator显示界面的逻辑中注册了登陆成功事件的监听

在Entry事件监听中处理MainSceneProxy实例的游戏数据的初始化，初始化MainSceneMediator管理的界面并显示

当MainSceneMediator初始化并显示时，获取MainSceneProxy实例并通过其提供的接口访问对应的游戏数据，将调用相应的接口改变当前游戏数据

当点击界面上的Enter按钮时，会派发Login事件和游戏模块的Entry事件

此时Login事件被MainSceneController监听，

在MainSceneController的Login处理逻辑中，获取到GameSceneProxy实例，并调用其提供的模拟登陆的接口login用于模拟登陆，5s之后会登陆成功

游戏界面模块处理Entry事件，处理游戏场景切换，将当前场景从主界面场景切换到游戏场景