**Unity游戏热更新接入**



**Version Control**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Change Description |
| 1.0 | 05/19/2016 | 邓世伟 | 初稿 |
| 1.1 | 05/19/2016 | 花成 | 增加LuaFramework框架及使用说明 |
| 1.2 | 05/20/2016 | 邓世伟 | 增加资源读取及部署 |
| 1.3 | 05/20/2016 | 花成 | 修改 框架结构 、项目示例 部分的内容 |



[Unity游戏热更新接入](#_Toc13155)

[Version Control](#_Toc8701)

[1 名词解释](#_Toc16592)

[2 AssetBundle打包及部署](#_Toc24040)

[2.1 打包规则](#_Toc11676)

[2.2 实施打包](#_Toc20091)

[2.3 打包生成文件](#_Toc27934)

[2.4 打包好的文件上传至服务器](#_Toc14820)

[3 资源读取及部署](#_Toc5171)

[3.1 读取配置](#_Toc21326)

[3.2 读取方式](#_Toc15385)

[4 文件部署](#_Toc7888)

[5 框架结构](#_Toc12561)

[5.1 AssetBundles文件夹](#_Toc22029)

[5.2 LuaFramework文件夹](#_Toc29456)

[5.3 LuaFramework/Lua文件夹](#_Toc23188)

[6 项目示例](#_Toc270)

[6.1 运行效果](#_Toc27778)

[6.2 Sample场景](#_Toc27568)

[6.3 创建UI界面 prefab](#_Toc20215)

[6.4 Lua脚本编写](#_Toc21089)

[6.4.1 新建视图及控制器脚本](#_Toc4435)

[6.4.2 增加全局变量定义](#_Toc17946)

[6.4.3 修改控制器管理器初始化](#_Toc11108)

[6.4.4 启动界面](#_Toc10075)



# 名词解释

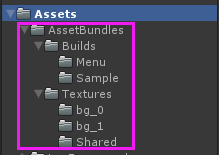
* 热更新（不停机状态下完全更新）
* AssetBundle（unity中的资源）

# AssetBundle打包及部署

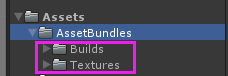
## 打包规则

把要打包的文件，放在Assets/AssetBundles跟目录下。具体规则如下：

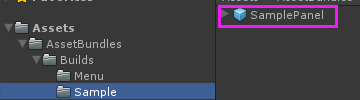
* 打包根目录的名称必须是AssetBundles



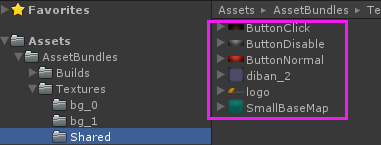
* 所有需要打包的文件放在Builds和Textures目录下



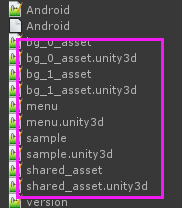
* Builds目录存放Prefab格式的文件



* Textures目录存放Prefab的依赖关系文件



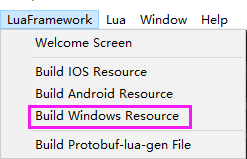
* Builds和Textures目录的文件夹根据项目需求随便命名，打包出来的assetbundle和该名称一样



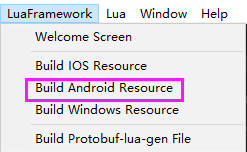
## 实施打包

选择需要打包的平台版本，进行相应的打包。具体如下：

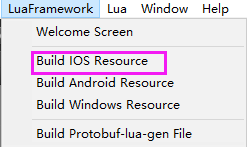
* Pc机器上的打包，点击Build Windows Resource



* Android设备上的打包，点击Build Adnroid Resource



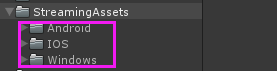
* IOS设备上的打包，点击Build IOS Resource



## 打包生成文件

打包生成的文件，存放在StreamingAssets目录中，具体如下：

* 根据不同的平台，生成的文件存放在不同的目录下



## 打包好的文件上传至服务器

打包好生成的文件，通过jenkins的方式，上传到远程服务器，服务器的目录结构和StreamingAssets目录一致



# 资源读取及部署

## 读取配置

资源可以通过AppConst.cs类，进行配置如何来读取不同地方的资源，具体如下：

* 读取本地的资源，设置DebugMode = true
* 读取Http服务器上的资源，设置UpdateMode = true和 DebugMode = false
* 读取本地打包好的资源，设置DebugMode = false 和UpdateMode = false

## 读取方式

打包好的资源，通过PanelManager.cs类，可以进行读取，具体如下：

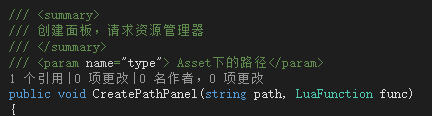
* 传入assetbundle的名称，进行读取,这种方式要求在打包的时候，assetbundle名称必须按规范的命名，命名规则“name”+“Panel”,如“SamplePanel”。读取资源的时候就只需要传入“Sample”即可





* 传入详细的assetbundle路径的形式，进行读取,如“Builds/Propmpt/Pro.png”这种方式没有命名规范要求，名称可以按自己意愿，如果代入后缀的话，可以读取到对应的类型资源。不代入的话默认\*.prefab





# 文件部署

要求Unity 5.0 以上版本。热更新框架已经发布为LuaFramewrok.unitypackage文件，方便项目组导入。具体流程如下：

* 在Unity Editor中新建项目或在已有项目中 导入LuaFramewrok.unitypackage文件
* 添加宏ASYNC\_MODE



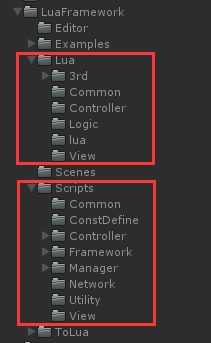
# 框架结构

## AssetBundles文件夹

存放需要打包的原始资源文件

## LuaFramework文件夹

主要是 Scripts文件夹和Lua文件夹，分别存放CS脚本文件和lua脚本文件



## LuaFramework/Lua文件夹

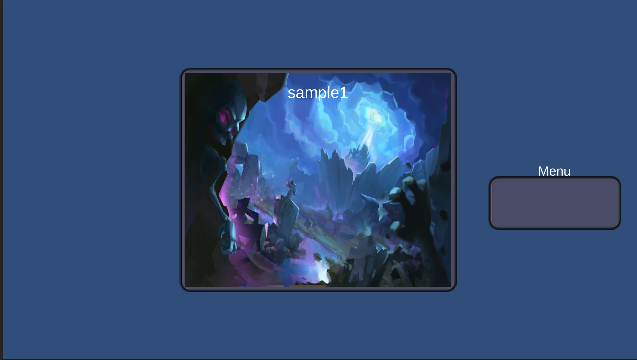
LuaFramework框架采用MVC模式，在 Assets\LuaFramework文件夹下有对应的文件夹

* Control\ 放置控制器文件
* View\ 放置界面视图文件
* Common\ 放置通用的全局定义文件

# 项目示例

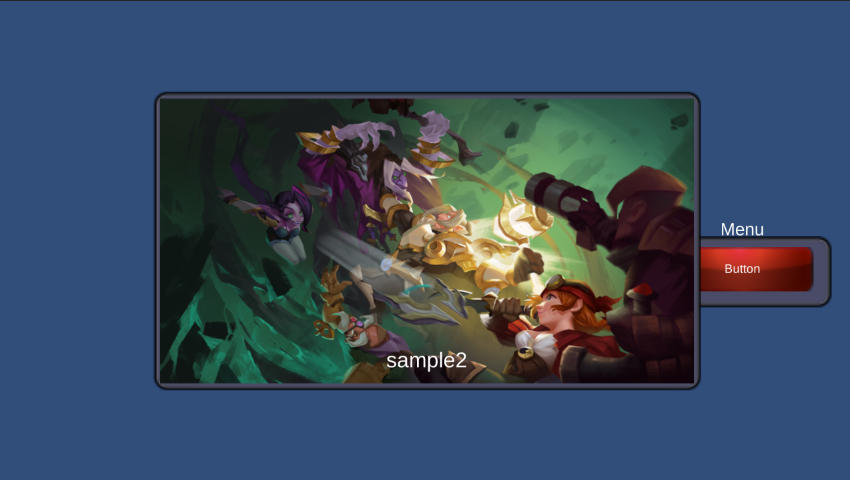
## 运行效果

第1版：服务端资源为第一版资源



游戏启动后通过 Lua脚本动态生成 SamplePanel 界面 和 MenuPanel界面。

第2版：服务端资源更新为第二版资源



SamplePanel，MenuPanel界面都会更新，MenuPanel界面会显示红色按钮，点击该按钮会切换隐藏/显示SamplePanel界面。

## Sample场景

在Assets\Scenes下已有一个Sample 场景文件，可直接使用此场景启动示例。

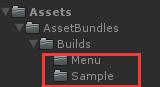
GameManager对象上已绑定Main.cs脚本，在Start()中发出StartUpCommand的消息来启动示例。

如果项目中有具体需求，可以覆写StartUpCommand的Execute方法。

## 创建UI界面 prefab

新建 MenuPanel.prefab 和 SamplePanel.prefa两个文件，

在Assetbundle\Builds文件夹下新建 Menu 和 Sample文件夹，分别放入这两个prefab文件。

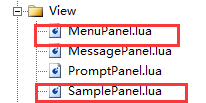


## Lua脚本编写

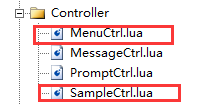
### 新建视图及控制器脚本

LuaFramework框架采用MVC模式，Sample不包含业务逻辑，Model部分先省略，编写View和Controller即可。

在LuaFramework\Lua\View下新建View脚本：MenuPanel.lua , SamplePanel.lua



在LuaFramework\Lua\Controller下新建Controller脚本：MenuCtrl.lua , SampleCtrl.lua



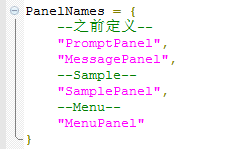
### 增加全局变量定义

修改Common\define.lua文件

加入控制器 的 KV对

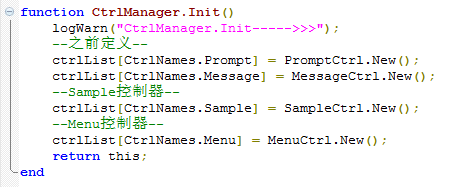


加入界面视图的枚举



### 修改控制器管理器初始方法

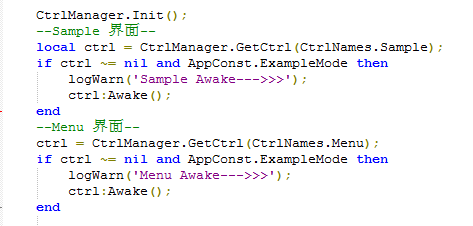
修改Logic\CtrlManager.lua，在CtrlManager.Init()中加入Samle Menu两个界面控制器



### 初始化界面

修改Logic\Game.lua文件

在 Game.OnInitOK()中 调用两个界面的控制器，生成界面



到此启动游戏后就会加载并显示这两个界面。

