# 打卡 写入版本信息

[202208281805](GIF/202208281805.gif)

# 打卡 写入AB包信息

[202208281811](GIF/202208281811.gif)

复制到外面[202208281919](GIF/202208281919.gif)

# 打卡 采用MD5，更新AB的变化

01 打一次包

02 修改标记要打包的某个资源

03 再打一次包

04 点击AB热更

05 可以看到更新AB的变化

[202209141635](GIF/202209141635.gif)

# 打卡 生成热更包配置表Patch

[202209141835](GIF/202209141835.gif)

# 服务器文件部署--------------------------

# Apache 下载

<https://httpd.apache.org/download.cgi#apache24>

[202209141859](GIF/202209141859.gif)

# 打卡 Apache 配置软件位置

[202209141908](GIF/202209141908.gif)

# bug 安装后不存在服务 服务开启失败

以下加颜色有用

查所有（管理员权限下cmd）

netstat -ano

查PID

tasklist |findstr 4

01 打开（或管理者运行）httpd.exe，窗口闪退

02 netstat -ano有端口占用情况，不知道是谁



03 改端口，再次打开依然闪退

D:\ProgramFilesTrim\Apache\Apache24\conf\httpd.conf

Listen 8081

...

ServerName localhost:8081

04 管理员权限cmd到bin目录，httpd.exe -k install

没用，这是个安装命令

05 管理员权限cmd到bin目录，

D:\ProgramFilesTrim\Apache\Apache24\bin\httpd.exe -k install -n “Apache2.4”

D:\ProgramFilesTrim\Apache\Apache24\bin\httpd.exe -k uninstall "Apache2.4"这个是卸载

双引号里面是什么，服务名也就是什么。但是服务启动不了

[202209142043](GIF/202209142043.PNG)

06 安装别的版本，走到生成服务时。mmp，80没解决，又来443，这么喜欢抢端口



07 杀死PID4，不能杀死

taskkill /PID PID /F

09 442(我的被占用了)全改443（我的这个没被占用）端口apache\conf\extra\httpd-ssl.conf

## 01 以上总结

01

改端口，再次打开依然闪退

D:\ProgramFilesTrim\Apache\Apache24\conf\httpd.conf

Listen 8081（我的这个没被占用）

...

ServerName localhost:8081

02

442(我的这个被占用了)全改443（我的这个没被占用，后面发现也被占用）端口apache\conf\extra\httpd-ssl.conf

03（这个关乎有没有服务）

管理员权限cmd到盘符（ProgramFilesTrim是不确定支不支持空格）

D:\ProgramFilesTrim\Apache\Apache24\bin\httpd.exe -k install -n “Apache2.4”

D:\ProgramFilesTrim\Apache\Apache24\bin\httpd.exe -k uninstall "Apache2.4" 这个是卸载



99 补充

（管理员权限下cmd）

查所有，netstat -ano

查PID ，tasklist |findstr 4

99 补充 后面出现了端口占用VSVN占用了443，结束掉了该进程（这玩意每次自启）（后来发现VSVN还会抢我Apache的端口，但也可能是时间久Apache关了）

99 补充，窗口又闪退，但是不影响，后台有进程



# 打卡 下载安装到打开apache首页

[202209142322](GIF/202209142322.gif)

# 了解 将AB变化移动的位置

D:\ProgramFilesTrim\Apache\Apache24\htdocs\AssetBundle

# 打卡 服务器配置表

复制 D:\Data\Projects\Unity\Ocean\_RealFram\_20220710\_2018.2.10f1\RealFram\_20220710\_2018.2.10f1\Hot\StandaloneWindows64\Patch.xml

到D:\ProgramFilesTrim\Apache\Apache24\htdocs\ServereInfo.xml

修改相关内容

[202209151420](GIF/202209151420.xml)

# bug AB总包名字不一样

打包后会自动生成StreamingAssets文件，但Ocean的叫StandaloneWindows64

所以把内部打包的位置加上平台名，导到外面，有趣的是，发现原来的StreamingAssets自动替换成StandaloneWindows64

[202209151413](GIF/202209151413.gif)

# 了解 下载的流程

## 工程的Hot文件夹

D:\Data\Projects\Unity\Ocean\_RealFram\_20220710\_2018.2.10f1\RealFram\_20220710\_2018.2.10f1\Hot\StandaloneWindows64



第一次全部打包，第二次修改某些文件（我选了image），再次打包出现的。反映打包的变化

StandaloneWindows64是文件夹的名字

**Patch.xml（自动）**

<?xml version="1.0" encoding="utf-8"?>

<Pathces xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" Version="1">

<Files Name="image" Url="http://127.0.0.1/AssetBundle/0.1/1/image" Platform="StandaloneWindows64" Md5="25925f17e67aaa08aa80dff09a1e87d5" Size="5.42773438" />

<Files Name="StandaloneWindows64" Url="http://127.0.0.1/AssetBundle/0.1/1/StandaloneWindows64" Platform="StandaloneWindows64" Md5="bd254a0eb7f6ea9ada10e8c9d03d556a" Size="1.75878906" />

</Pathces>

## Apache

D:\ProgramFilesTrim\Apache\Apache24\htdocs

index.html自带的

其它都要自己创建（根据Hot文件夹的内容，复制过来）



D:\ProgramFilesTrim\Apache\Apache24\htdocs\AssetBundle\0.1\1



Apache服务器弄好后，可以访问（8081是D:\ProgramFilesTrim\Apache\Apache24\conf\httpd.conf照片呢个设置的，之前有说）



**ServerInfo.xml（手动复制Patch修改）**

<?xml version="1.0" encoding="utf-8"?>

<ServerInfo>

<GameVersion Version="0.1">

<Pathces Version="1" Des="测试热更">

<Files Name="image" Url="http://127.0.0.1/AssetBundle/0.1/1/image" Platform="StandaloneWindows64" Md5="25925f17e67aaa08aa80dff09a1e87d5" Size="5.42773438" />

<Files Name="StandaloneWindows64" Url="http://127.0.0.1/AssetBundle/0.1/1/StandaloneWindows64" Platform="StandaloneWindows64" Md5="bd254a0eb7f6ea9ada10e8c9d03d556a" Size="1.75878906" />

</Pathces>

</GameVersion>

</ServerInfo>

## PC模拟下载的地方

C:\Users\lenovo\AppData\LocalLow\DefaultCompany\RealFrame（RealFrame是PlayerSettings设置的项目名）

主要验证ServerInfo.xml有没有从Apache中更新过来

在HotPatchMgr.cs中，我用第一句得不到，用第二种的得到了（在工程中报的错误是xml转ServerInfo，因为xml为空）

//private string m\_hotCfg = "http://127.0.0.1/ServerInfo.xml"; //热更配置表

private string m\_hotCfg = "http://localhost:8081/ServerInfo.xml";



## 建议 做时用快捷方式



# bug Editor文件夹外不能使用

打包时，第一句报错，第二句可以

就是打包时，会检查项目编译，发现不在Editor的文件使用了EditorUserBuildSettings.activeBuildTarget.ToString()，会报错、

暂时是打包之前去设置一个字符串

EditorUserBuildSettings.activeBuildTarget.ToString()

StandaloneWindows64

# bug 校验码错误，重新下载4次仍失败

7e,b3是代码中MD5的最后两位数字

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | new | old | local | server |  |
| Apache |  |  |  |  | b3 |
| PC下载 |  |  | 5a | b3 |  |
| 代码 | 7e | b3 |  |  |  |

|  |  |
| --- | --- |
| m\_DownLoadMD5Dic | downLoadAssetLst |
| b3 |  |
| Url "http://127.0.0.1/AssetBundle/0.1/1/image" | downLoad.SaveFilePath = "C:/Users/lenovo/AppData/LocalLow/DefaultCompany/RealFrame\_Test/DownLoad/image" |

加黄有问题，

我改成

void AddDownLoadList(Patch patch)

{

patch.Url = patch.Url.Replace("http://127.0.0.1","http://localhost:8081");//http://127.0.0.1==http://localhost

后来想想8081是当做服务器的，不能直接用

得要开启能访问http://127.0.0.1的服务

**管理者cmd sc config http start= demand & net start http**





# 打卡 从Apache下载到本地

01 打一次全包（之前打过就不用了）

02 热更编辑器打一次（之前打过就不用了）

03 修改某个资源，再打一次包

04 热更编辑器再打一次。得到（圈圈是固定的）

（在D:\Data\Projects\Unity\Ocean\_RealFram\_20220710\_2018.2.10f1\RealFram\_20220710\_2018.2.10f1\Hot\StandaloneWindows64）



05 将第一第二个复制到

D:\ProgramFilesTrim\Apache\Apache24\htdocs\AssetBundle\0.1\1

06 将Patch修改D:\ProgramFilesTrim\Apache\Apache24\htdocs\ServerInfo.xml

内容看以前的“了解 下载的流程”

07 进行测试（要复现这个场景，需要重复03-06，或者删除"C:/Users/lenovo/AppData/LocalLow/DefaultCompany/RealFrame\_Test/DownLoad/”文件夹）

[202209202025](GIF/202209202025.gif)

# 了解 80与8081；localhost与127.0.0.1

http://127.0.0.1/ServerInfo.xml（默认是80端口，所以等于http://127.0.0.1:80/ServerInfo.xml）

但是Apache改成8081后，这里也要改成

<http://127.0.0.1:8081/ServerInfo.xml>

或者localhost:8081/ServerInfo.xml

具体位置是生成热更包Editor和热更Manager

# 打卡 从Apache下载到本地

[202209271750](GIF/202209271750.gif)

# 打卡 热更包

可以看到image和64这两个包的修改时间是58分（其他的是50分）

[202209271800](GIF/202209271800.gif)

当我把C:\Users\lenovo\AppData\LocalLow\DefaultCompany\RealFrame\_Test\Download中用到的包删除，会报错。也就是AB包内部会根据打包时的目录形成一个局部目录结构。注意的事，不要因为AB包中的路径与unity的资源路径一样，就以为加载AB时用的是unity中的资源。

[202209282218](GIF/202209282218.gif)

# bug 下载到本地后

本地的，和assetbundleconfig认定的untiy中打包的路径是不一样的，那么crc也是不同的，就拿到不到resItem

因为尝试强制设置一个crc是可以拿到resItem的

[202209272105](GIF/202209272105.PNG)

以上的影响就是：实际的打包的m\_Path要统一（Ocean可能后面会统一）

## 了解

043-下载测试 10：59（括号内是给一个关键字命名）

Ocean没有引用PC模拟下载的地方C:\Users\lenovo\AppData\LocalLow\DefaultCompany\RealFrame\_Test\Download（Local）

而是复制了外部的AB包D:\Data\Projects\Unity\Ocean\_RealFram\_20220710\_2018.2.10f1\RealFram\_20220710\_2018.2.10f1\AssetBundle（Outter）

到内部D:\Data\Projects\Unity\Ocean\_RealFram\_20220710\_2018.2.10f1\RealFram\_20220710\_2018.2.10f1\Assets\RealFrame\StreamingAssets\StandaloneWindows64（Inner）

# 打卡 44 AB包加载 热更回退

**热更回退的代码逻辑**

GameStart Open（HotfixWnd）//打开热更面板

HotfixWnd(Hotfix())//热更检查

AssetBundleMgr//中有路径判断是加载热更的还是本地的

**操作流程**

全打AB包一次，作为本地资源，暂时复制一份，演示时放到StreamingAsset中作为本地ab加载.

修改图片，再打AB包，再打热更AB包，部署到Apache

Ocean的是，热更的脸是花的，干掉就加载不花的

我的是，老的是全红的，新的（热更）的有一杠黑。从黝黑到全红，演示了热更回退

[202209292338](GIF/202209292338.gif)

# 打卡 AES加密（写死是某个文件）

[202209301005](GIF/202209301005.gif)

# 了解 FileStream

FileStream类型对象是引用类型，并且作为参数传参时，可以不用ref returm out来获取变化

# 打卡 AES解密（写死是某个文件）

[202209301417](GIF/202209301417.gif)

# bug System.Security.Cryptography.CryptographicException: Padding is invalid and cannot be removed.

fs.Seek(0, SeekOrigin.Begin); //读完 seek回去，保持原始状态

上面的，放1的后面就报错，2的前面就正常

[202209301419](GIF/202209301419.PNG)

# 打卡 AB包的加密解密（AES）

[202209301523](GIF/202209301523.gif)

# 打卡 删除外部AB路径中的.manifest

[202209302008](GIF/202209302008.gif)

然后加密，为后面加载解密AB做准备。

我把操作分开写，因为我打好了包，不想重跑一遍，所以增量用MenuItem

# 了解 安卓下不能FileStream 读取AES加密的AB

所以要解压

IOS。PC则不用加压

# 了解 C# 类中属性

编辑器面板不显示

# 打卡 WebRequest下载加密包到本地的Origin

手动//把加密的ab复制到内部StreamingAssets（节省上传服务器再下载）

HotPatchMgr.Unpack2PersistentDataPath//解压（加密的ab）到Application.persistentDataPath + "/Origin";

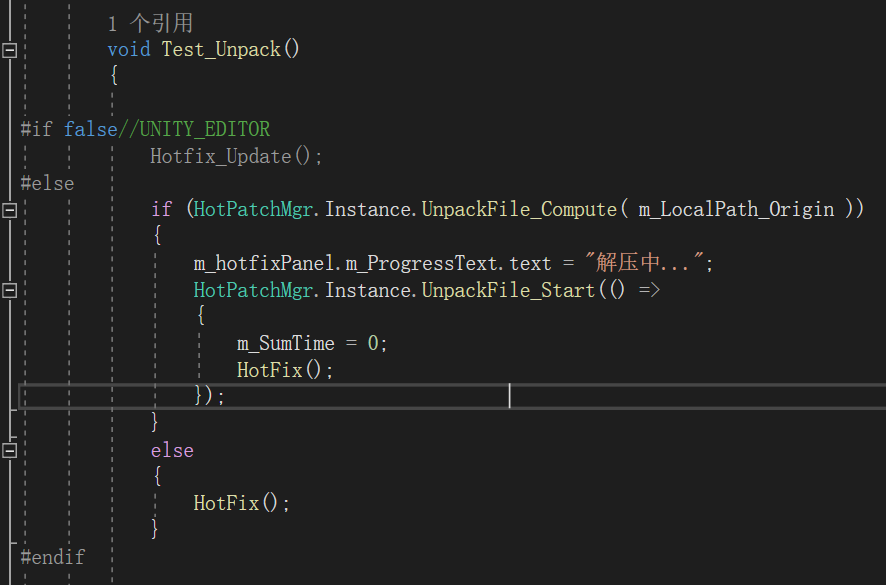
[202210020018](GIF/202210020018.gif)

后面报的是Stream中不允许使用FileStream的错误，后面再处理

# 打卡 解压热更及加载测试

## 01

下载后的包是不是解密的（测试是加密的，跟从Apache下载到DownLoad是一样的原理。这里是用WebRequest从StreamingAssets下载加密包到本地的Origin）。注意HotfixWnd中，走下面的代码去解压

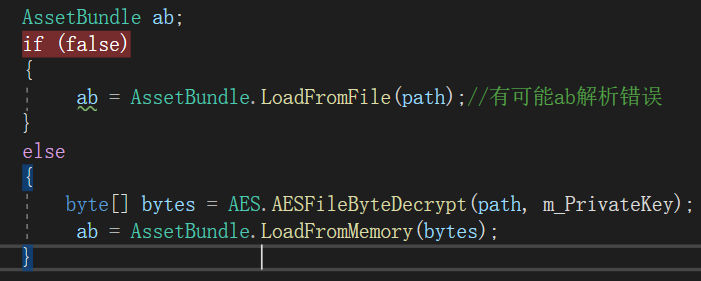


## 02

使用资源（图片，预制体）时，ab包的加载。

看到ABMgr中，参照abCfg的加载有AssetBundle.LoadFromFile和AssetBundle.LoadFromMemory

aABMgr.LoadABItem中也修改。



## 03

加载到了，判断是不是解压的，看其中的脚本图标。

但我加载的是图片，致命的是前期的Mono脚本挂预制体报错的问题没解决。

得回去让它能挂载脚本（也不知道为什么挂在脚本没报错了）

[202210022039](GIF/202210022039.gif)

# ILRuntime----------------------

# 了解 ILRuntime

JIT == dll替换

IOS不支持dll替换

官网

https://ourpalm.github.io/ILRuntime/public/v1/guide/index.html

官方GitHub

https://github.com/Ourpalm/ILRuntimeU3D/

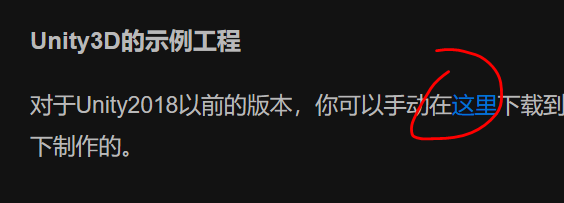
推荐的博客

https://www.cnblogs.com/zhaoqingqing/archive/2019/01/17/10274176.html

# 下载 示例工程

（还有一个比较复杂的源工程，不用它）

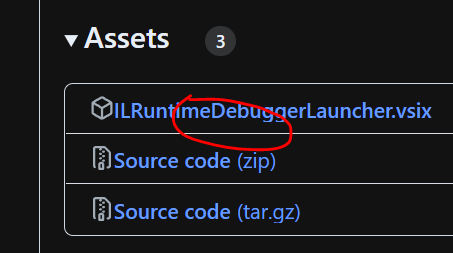
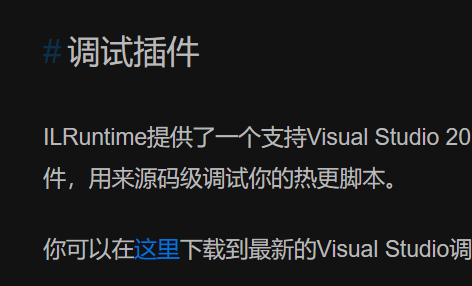
https://ourpalm.github.io/ILRuntime/public/v1/guide/tutorial.html



# 下载 调试插件

双击安装

后面勾选2017（2019没试过）[202210022207](GIF/202210022207.PNG)



# 打卡 添加mcs.rsp

[202210022213](GIF/202210022213.gif)

实例有C++指针，我们的工程要添加unsafe的安全访问文件夹

出处

<https://ourpalm.github.io/ILRuntime/public/v1/guide/tutorial.html>

# 打卡 复制三个文件夹（版本确认）

ILRuntime

Mono.Ceil.20

Mono.Ceil.Pdb

到RealFrame中，但是这版本的ILRuntim没有这3个（因为迭代了）

视频里是2018年5月2日的commits 1f47732b，版本是v1.6.2

但是，用友以上文件的只有v1.3.0，并且v1.3.0是最初的版本，下一个版本就是v1.6.2

[202210031549](GIF/202210031549.PNG)

所以最终下载v1.3.0

版本还是一致的好，根据新版本自己来调整不是新手该做的事

[202210031555](GIF/202210031555.gif)

# 了解 github下载历史版本

[202210031527](GIF/202210031527.gif) 或者

[202210031552](GIF/202210031552.gif)

# 打卡 4个报错全注释掉

[202210031559](GIF/202210031559.gif)

# 打卡 新建VS类库工程准备（推荐是2017-2019）

## 01 HotFixProject下HotFix类库工程

用的是vs2017

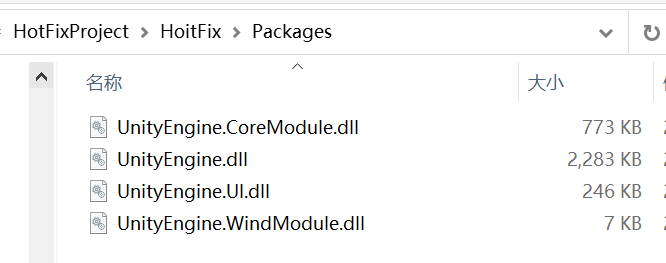
类库，Framework4.6

[202210031626](GIF/202210031626.PNG)

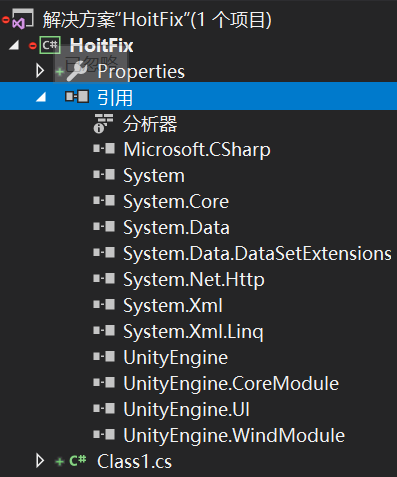
## 02 引入Unity软件的dll

到对应版本的Untiy路径下搜，同名很多歌，引用错误就换

第一个选最大的那个，22M左右（其它看着大小找）我复制放的位置是



引用



## 03 引入RealFrame的dll

看自己怎么划分程序集，基本是CSharp+所有自己创建的程序集

这些经常变化，直接引用到它们的初始路径

类似D:\Data\Projects\Unity\Ocean\_RealFram\_20220710\_2018.2.10f1\RealFram\_20220710\_2018.2.10f1\Library\ScriptAssemblies

[202210031736](GIF/202210031736.PNG)

## 04 全部引用设置“不复制”

[202210031741](GIF/202210031741.gif)

## 05 输出路径

[202210032231](GIF/202210032231.PNG)

D:\Data\Projects\Unity\Ocean\_RealFram\_20220710\_2018.2.10f1\RealFram\_20220710\_2018.2.10f1\Assets\RealFrame\GameData\Data\HotFix

# bug 类型 Debug同时存在于UnityEngine.CoreModule UnityEngine

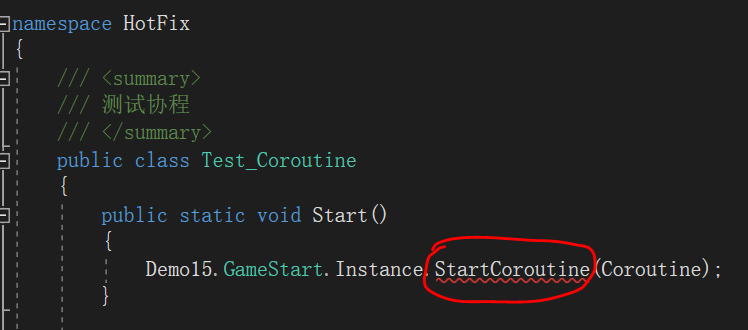
（但是773和22M的有冲突）

我取消引用CoreModule（因为它小，有问题以后再处理）

[202210032213](GIF/202210032213.PNG)

后续来了，使用不了StartCoroutine。

尝试删除22M的UnityEngine.dd，保留773k的CoreModule.dll，没报错



# 打卡 第一个方法Debug.Log

[202210032227](GIF/202210032227.PNG)

点击生成（可以看到命名错了，后面会改回来）

[202210032314](GIF/202210032314.PNG)

打包时用到pdb dll，但Untiy打包是不打包这些，所以加后缀.txt（后面改名字又生成一次，发现mdb没了）

[202210032316](GIF/202210032316.PNG)

# 打卡 AB设置

[202210032328](GIF/202210032328.PNG)

再打一次外部包

[202210032335](GIF/202210032335.PNG)

# 打卡 ILRuntimeMgr与目录结构

[202210032353](GIF/202210032353.PNG)

我目录不一样，所以加了程序集、引用、**允许unsafe**

让新建的ILRuntimeMgr（属于Manager）能引用ILRuntime（新建程序集ILRuntime）的代码

# bug AssetBundle.LoadAsset失败

01 回溯起来是，我把m\_AppDomain局部参数了（在方法内部实例化）。也行，但要加ref 或 out

02 类库工程应用，我移除了Demo相关的（不知道没有这因素）

# 了解 VS工程改名后没生成mdb

首次生成了，后面没生成，但没影响

# 打卡 ILRuntime引用void方法

总结

一个类库工程写了方法

生成dll到unity中，全部加后缀.txt

执行一次内部打包（详细看AssetBundleMgr.InitMgr）

然后在ILRuntime中调用

[202210041920](GIF/202210041920.gif)

# 打卡 小工具+”.txt”

[202210041956](GIF/202210041956.gif)

# 注意 不使用AB包

此时频繁变更dll，先不要进行AB包加载的方式

# 打卡 ILRunTime之1个参数的方法

[202210042257](GIF/202210042257.gif)

# 打卡 ILRunTime之参数数量不匹配

[202210042300](GIF/202210042300.gif)

# 打卡 ILRunTime之又一种1个参数的方法

[202210042309](GIF/202210042309.gif)

# 打卡 ILRunTime之无参构造与有参构造

初始值0

无参构造改值100

有参构造改值15

打印

[202210050111](GIF/202210050111.gif)

# 打卡 ILRunTime之两种有参泛型方法

[202210050229](GIF/202210050229.gif)

# 打卡 ILRunTime之两种有参委托（后面加了Action）

[202210051424](GIF/202210051424.gif)

# 打卡 ILRunTime之跨域委托调用（系统自带）

不是系统自带的适配错误

[202210051529](GIF/202210051529.gif)

注释掉不是系统自带的

[202210051533](GIF/202210051533.gif)

# 注意 每次的操作

生成

到Unity中点击“后缀加.txt”的操作（前面有讲到）

# 打卡 ILRunTime之跨域委托调用（自定义）

但官方说不推荐使用

具体方法太长，详细看代码

[202210051707](GIF/202210051707.gif)

# 打卡 ILRunTime之跨域委托调用（热更域定义，主程调用）

[202210051751](GIF/202210051751.gif)

# 打卡 ILRunTime之两种跨域继承的方式

[202210052329](GIF/202210052329.gif)

# bug 编辑器损坏

generated可能有一些关于Transform等的映射，我在这个文件夹下新建程序集generated想导出出dll给Mgr使用，结果导出了一大堆玩意（名字是Unity各种组件和包，这个就是dll属性复制路径要设置为false），删完后编辑器就坏了。

只能回退和重新安装

这一回退，把之前写的笔记弄没了（还好类都弄到Mgr\_Data中了）。但是新建的记录图还在

# 了解 引用关系

ILRuntimeMgr(简称ILRMgr)

ILRMgr和相关类全部放ILRuntime中，去引用Manage,Common,Data

因为我不想把ILRun整个文件夹杂糅在Manager或CSharp里面

所以将ILRMgr和相关类从原先的Manager中拎出去到ILR文件夹

但是逻辑上讲ILRMgr，放Manager是比较合理的。看后续有没有时机（知识储备和时间）改。

个人觉得不使用程序集还是不要的好，程序集检验了代码的解耦性。

# 了解ILRuntimeCLRBinding.cs

Assets/ILRuntime/ILRuntime/Adapters/Editor/ILRuntimeCLRBinding.cs

自动生成绑定脚本到Assets/ILRuntime/ILRuntime/Generated

编辑器脚本

依赖于ILRMgr中的类

我会整理好它

# 打卡 ILRunTime之自动Binding的使用和效率

取消注释，Assets/ILRuntime/ILRuntime/Adapters/Editor/ILRuntimeCLRBinding.cs。之前有注释掉的地方

[202210071152](GIF/202210071152.gif)

可以看到，执行100000次加法

没绑定，6点多毫秒

绑定，2点多毫秒

[202210071227](GIF/202210071227.gif)

也就是绑定的话，速度更快

# 打卡 ILRunTime之协程

[202210071719](GIF/202210071719.gif)

# bug 程序集引用

ILRuntimeMgr（属于ILRuntime）引用GameStart（属于Demo15）

用dll回重复报错

所以GameStart中调用ILRuntimeMgr.InitMgr(gameObject)，让ILRuntimeMgr用到GameStart.Instance.gameObject

但是，最后还是把ILRuntime、ILRuntimeMgr、其它Manager和热更案例脚本全部放一起了（ILRuntimeMgr还要用到ResourceMgr）

# 打卡 ILRunTime之重定向（AddCompontent）

有生命函数的输出说明Add上了

[202210072254](GIF/202210072254.gif)

GameStart所在节点会有适配器

[202210072319](GIF/202210072319.PNG)

# 打卡 ILRunTime之重定向（GetCompontent）

[202210072322](GIF/202210072322.gif)

# 了解 ILRuntime状况

最新的，友好支持项目后期的热更

https://github.com/wuxiongbin/XIL

# 了解 自底向上访问

委托，命令，事件

# 了解 消息机制

01 transform.Find(“”)

02 委托，SendMessageUpward(“”)

03 unity自带的消息机制，自定义消息机制

04 消息机制

# 打卡 ILRuntime之根据resource=false（相当于ILR热更）来使用AppDomain，用于调用Window的生命函数

01 将LoadingUi,MenuUi添加到类库工程（最新的工程没有这两个脚本，我将demo14的两个wnd合成一个作为Dem015，但是现在需要用两个的来实现ILR，复制一份Demo16吧）

02 就是查看Window的生命函数，然后照着改（我已改）

03 DelegateManager.RegisterMethodDelegate，只有1,2,3,4个参数的重载，照着改一个5个参数的，并且修改相关（MethodDelegateAdapter）

04 为ResourceMgr.OnAsyncObject(Ocean的命名可能被我改了)的委托参数做ILR注册

05 WindowAdapter在ILRuntimeCLRBinding.cs进行注册，在ILRuntimeMgr进行绑定

06 生成dll，gaichengtxt，生成绑定脚本

07 到UIMgr.OpenWNd修改ILR热更部分

## 了解 示例运行流程

按钮，点击打开LoadindWnd，执行生命函数，进行场景切换，其中又打开了MenuWnd

## bug CoreModule和UnityEngine冲突

导致Sprite等报错，同时存在于以上两个。暂时只采用打印来验证。

## 打卡 ILR和本地

切换为ILR

[202210111526](GIF/202210111526.gif)

切换为本地

[202210111527](GIF/202210111527.gif)

# bug TypeLoadException: Cannot find Adaptor for:Window

比较一个通过的Adapter与原类的区别

最后发现没注册在ILRuntimeMgr

# bug UnityEngine.UI.dll image.sprite找不到

CoreModule冲突

# 了解 用字符串的方式用到方法

Window，有时用字符串的方式用到它方法，比如”OnAwake”，但有时想改方法名就麻烦，所以静态字符串来引用

[202210101336](GIF/202210101336.PNG)

# 打卡 解决热更工程dll冲突

exe要运行，所以想解决掉dll冲突(2018.2.10f1)

## UntiyEngine.dll

2283K D:\Program Files\Unity\Hub\Editor\2018.2.10f1\Editor\Data\Managed

74K D:\Program Files\Unity\Hub\Editor\2018.2.10f1\Editor\Managed\UnityEngine

66K D:\Program Files\Unity\Hub\Editor\2018.2.10f1\Editor\Data\PlaybackEngines\windowsstandalonesupport\Managed

66K D:\Program Files\Unity\Hub\Editor\2018.2.10f1\Editor\Data\PlaybackEngines\windowsstandalonesupport\Variations\win32\_development\_mono\Data\Managed

66K D:\Program Files\Unity\Hub\Editor\2018.2.10f1\Editor\Data\PlaybackEngines\windowsstandalonesupport\Variations\win32\_nondevelopment\_mono\Data\Managed

66K D:\Program Files\Unity\Hub\Editor\2018.2.10f1\Editor\Data\PlaybackEngines\windowsstandalonesupport\Variations\win64\_development\_mono\Data\Managed

66K D:\Program Files\Unity\Hub\Editor\2018.2.10f1\Editor\Data\PlaybackEngines\windowsstandalonesupport\Variations\win64\_nondevelopment\_mono\Data\Managed

## CoreModule.dll

866K D:\Program Files\Unity\Hub\Editor\2018.2.10f1\Editor\Managed\UnityEngine

758K D:\Program Files\Unity\Hub\Editor\2018.2.10f1\Editor\Data\PlaybackEngines\windowsstandalonesupport\Managed

758K D:\Program Files\Unity\Hub\Editor\2018.2.10f1\Editor\Data\PlaybackEngines\windowsstandalonesupport\Variations\win32\_development\_mono\Data\Managed

758K D:\Program Files\Unity\Hub\Editor\2018.2.10f1\Editor\Data\PlaybackEngines\windowsstandalonesupport\Variations\win32\_nondevelopment\_mono\Data\Managed

758K D:\Program Files\Unity\Hub\Editor\2018.2.10f1\Editor\Data\PlaybackEngines\windowsstandalonesupport\Variations\win64\_development\_mono\Data\Managed

758K D:\Program Files\Unity\Hub\Editor\2018.2.10f1\Editor\Data\PlaybackEngines\windowsstandalonesupport\Variations\win64\_nondevelopment\_mono\Data\Managed

## 总结

可以看到，有6种路径。尝试控制变量法，我在02路径（01不行）用到UnityEngine，在01路径用到CoreModule，AudioModule，解决

[202210121718](GIF/202210121718.gif)

# Bug Untiy VS有时Ctrl+F5不起来

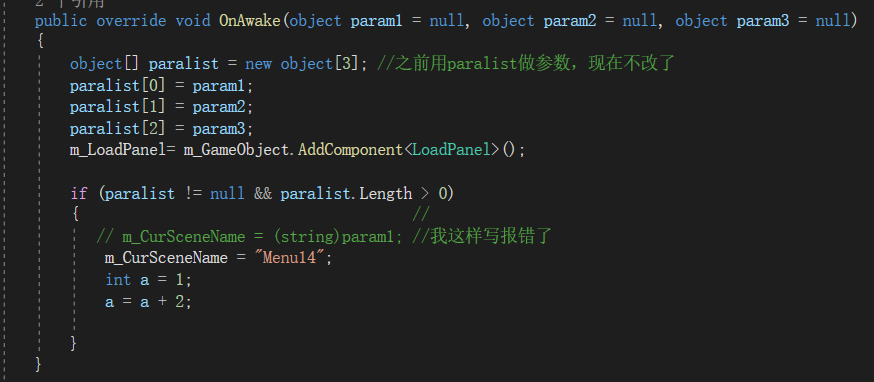
# 了解 命名

对UIMgr采用了以下的命名（容易找）

命名，主体\_对它的操作

# bug (string)object InvalidCastException: Specified cast is not valid.

暂时直接写字符串，因为有命名空间限制住，可以这样写



# Jenkins--------------------------

没怎么看懂，好像是没有push SVN之前，用Jenkins打包会复原到上次push的状态

revert

Demo16为例

# 了解 打包后看日志的位置

C:\Users\lenovo\AppData\LocalLow\DefaultCompany\RealFrame\_Test\output\_log.txt

# （未 先跳过这几集，暂时卡在svn不是内外部命令）打卡 本地打包运行测试

01

打包时要处理StreamingAssets\StandaloneWindows64中的StandaloneWindows64

我直接写在Common中

02

需要打AB包放在打包的文件夹\RealFrame\_Test\_Data\RealFrame\StreamingAssets\StandaloneWindows64

# bug 'svn' 不是内部或外部命令，也不是可运行的程序或批处理文件。

安装svn的时候command line client没选中安装

本地环境变量是否配置了

# 了解 SVN汉化

<https://tortoisesvn.net/downloads.html>

01 版本对应

02 msi放到根目录

03 删除根目录的Language里面的所有东西

我是卡在03上

# 打卡 RealFrameCfg.asset增加ProtobufPath

[202210131633](GIF/202210131633.gif)

# protobuf-net---------------------------------

曾用地址

https://github.com/mgravell/protobuf-net

<https://github.com/protobuf-net/protobuf-net>

版本2.4.0

[202210131558](GIF/202210131558.PNG)

下载的位置

C:\Users\lenovo\.nuget\packages

将C:\Users\lenovo\.nuget\packages\protobuf-net\2.4.0\lib\net40\protobuf-net.dll（269K）放到unity工程的plugins目录

# 打卡 Xml2Protobuf

Xml2Protobuf，就是Xml2Bin(bytes)，只是类要用Protobuf标签

01 xml名，必须有对应的名字的类

02 对类和类中用到的类都加上Protobuf的标签（比如MonsterData中用到了List<MonsterBase>）

03 对类和类中用到的类的字属（字段属性）都加上 [ProtoMember(1)]（从1开始，依次增加）【 [XmlIgnore]这种就不加】

可以看到MonsterData.xml转Bin，2k；转protobuf，1k

[202210132146](GIF/202210132146.gif)

# 打卡 Protobuf2Class（测试）

[202210132219](GIF/202210132219.gif)

# 打卡 Protobuf之继承类

[202210132229](GIF/202210132229.gif)