# 打卡 新建脚本就有了默认程序集

[202207102242](GIF/202207102242.gif)

# 打卡 根目录新建程序集就抢了默认程序集的地位

[202207102244](GIF/202207102244.gif)

# 打卡 子目录（不在根目录）新建程序集，默认程序集又有地位了

[202207102246](GIF/202207102246.gif)

# 打卡 程序集在VS中的显示

[202207102256](GIF/202207102256.gif)

# 打卡 程序集面板

[202207111039](GIF/202207111039.PNG)

# 加载方式------------------

# 01 拖

# 02 Resources

上限制2G

AssetBundle加载更快

# 打卡 03 AssetBundle 04AssetDataBase

[202207111217](GIF/202207111217.gif)

# 打卡 Xml2Class

[202207111341](GIF/202207111341.gif)

# 打卡 Class2Xml

[202207111352](GIF/202207111352.gif)

# 打卡 Class2Bin

[202207111432](GIF/202207111432.gif)

# 打卡 读取Asset(里面是ScriptableObject)

[202207111505](GIF/202207111505.gif)

# AB-----------------------------

# 打卡 配置一个ABConfig

[202207111902](GIF/202207111902.gif)

# 打卡 打包Prefab

太快，闪过去了

[202207111945](GIF/202207111945.gif)

# 打卡 标记 命名需要AB的文件

[202207121028](GIF/2022071121028.gif)

# 打卡 清理标记 清理命名的AB文件

[2022071221045](GIF/202207121045.gif)

# 打卡 打包标记的文件

[2022071121140](GIF/202207121140.gif)

# bug The name `AssetDatabase' does not exist in the current context

之前也没报错

暂时注释掉

# 打卡 删包

[202207121259](GIF/202207121259.gif)

# 打卡 生成xml（可视化），Bin（读取）

[202207121609](GIF/202207121609.gif)

# 打卡 只写出有效路径

## 写之前

[202207121613](GIF/AssetBundleConfig.xml)

## 写之后

[202207121631](GIF/AssetBundleConfig 之后.xml)

[202207121633](GIF/202207121633.gif)

# bug Bin文件放在StreamingAssets报Null

先放根目录

做成AB，再加载

# bug SerializationException: Could not find type 'System.Collections.Generic.List

我把ABConfig放Editor

# 注意 ABConfig

Demo01-04有一个

Editor和Demo05有一个

不一样的

我用程序集

# bug 加载的预制体丢失贴图 shader hiden internalError

代码控制正常

[202207130128](GIF/202207130128.txt)

# bug 生成xml为空

生成代码顺序

# 打卡 游戏中加载AB中的prefab

[202207130130](GIF/202207130130.gif)

# 注意 Mono中有AssetDataBase，打包时会报错，打包前注释掉

# 打卡 将生成的配置Config，做成ab包来加载

[202207130154](GIF/202207130154.gif)

# 打卡 通过Mgr加载AB

[202207131605](GIF/202207131605.gif)

# 打卡 字典的get set

[202207141035](GIF/202207141035.gif)

# 打卡 演示同步资源的加载和卸载

[202207141652](GIF/202207141652.gif)

# 打卡 演示异步资源的加载和卸载

[202007150130](GIF/202207150130.gif)

# 打卡 是否预加载的速度差距

因为刚停止，马上再运行。与隔了很久才运行是有差别的。

所以预加载与否、以上两种情况，一共演示四种情况

[202207151354](GIF/202207151354.gif)

# 了解 写代码习惯

经常开头renturn掉不对的条件

[202207152115](GIF/202207152115.PNG)

# 了解 数据结构

crc，与path互转，索引AB包

GUID，唯一标识符，GetInstaniateID()；AB也有，所有在Unity中的资源都有，但ABItem不存储该字段

ref，引用计数，ABItem,ResItem有，因为ABItem被ResItem引用，ResItem被ResObj引用，被引用就计数

JmpClr，跳转场景是否清除缓存，ResItem和ResObject在内存中，而ABItem是数据来源

ABItem,ResItem活跃在AssetBundleMgr,ResourceMgr，ResObj活跃在ObjectMgr

[202207171424](GIF/202207171424.PNG)

# bug 修改脚本名导致bytes不可用，解析不了

在打包2后，修改了涉及bytes的脚本名字。导致bytes废弃。

而要再次拿到可用bytes

01 恢复打包1的代码，包括隔离生成原生bytes的代码

02 修改bytes中被涉及到的部分

03 改回原来的代码名



# 了解

运行后退出，隔一会，再运行，不然上次的内存来不及清理

# bug 清存后不能深度清存

[202207181501](GIF/202207181501.gif)

RefCnt函数正负放了，导致RefCnt>0，请不了

# 打卡 同步加载的清存（缓存）、深度清存（不缓存）

实例的任务Texture2D变化大，所以重点看Texture2D

[202207181632](GIF/202207181632.gif)