# 资源框架

资源框架分为ResourceManager，AssetBundleManager，ObjectManager三部分；游戏初始化在空对象挂载一个脚本RFramework（继承可挂载单例，负责管理类的初始化），以此初始化ResourceManager，AssetBundleManager，ObjectManager；

ResourceManager的初始化，根据优先级的数量，New出异步加载资源队列的数量（一个优先级一个List），方法传入挂载空对象的MonoBehaviour，用空对象的MonoBehaviour开携程，死循环，按照优先级，先下载当前最高优先级的资源的加载列表List，编辑器下直接根据路径用AssetDatabase下载（非编辑器模式，调用GetCacheResouceItem方法从字典缓存中获取，更新上次使用时间和引用计数，如果缓存中没有调用abManager的LoadResouceAssetBundle方法，根据crc值加载ab包，根据资源名字m\_AssetName，用ab包异步加载的方式加载出ab包，更新上次yeild的时间）；调用CacheResource方法缓存起来（调用WashOut方法，如果超出最大限制，清除最早未使用的资源，重新赋值刷新资源的引用次数和上次使用时间，存进字典）；调用自己异步加载结束的回调，重置回调方法，并且回调池回收这个回调，如果上次yeild时间超过最大加载时间，则进行yeild操作，进行其他操作，防止线程阻塞，等待下次调用继续下载。

AssetBundleManager的初始化LoadAssetBundleConfig，调用AES类AESFileByteDecrypt解密的方法，传入路径和秘钥，把该路径下的文件读取字节流，截取头标签和AES定义的头标签验证解密的是否是AES文件（防止浪费性能并且提高了安全性），是的话开始解密，返回解密成功的字节流数据。调用AssetBundle.LoadFromMemory（解密的字节流数据）获得清单列表的ab包，获取清单ab中的TextAsset，转为字节流内模式在序列化为自定义的 AssetBundleConfig类，遍历获得AssetBundleConfig类中ABBase的集合，把ABBase的值赋给ResouceItem类，并且以crc作为key，ResouceItem作为值存进字典中。

ObjectManager的初始化Init（回收池节点，场景节点），调用GetOrCreatClassPool<ResouceObj>(1000)方法，创建资源对象的对象池，如果类对象池字典中有，直接返回这个对象池，没有的话，New一个对象池类，存入类对象池字典并且返回这个对象池。

## 辅助类：

1. ResouceObj（单个对象类）
2. ResouceItem（单个对象的ab包类）
3. AssetBundleConfig清单文件类：可序列化，存储自定义ABBase的集合
4. ABBase（ab包基类）：可序列化（中间类，序列化后深拷贝赋值给ResouceObj类）
5. AsyncCallBack（回调类，存结束的委托和调用委托参数和重置方法（重置后可用对象池回收））
6. AsyncLoadResParam（异步加载对象类，存储加载完成后的回调集合）

## 辅助枚举：

LoadResPriority（加载资源优先级）：mid，low，hight，num

## ResourceManager

## AssetBundleManager

**字段：**

m\_ResouceItemDic（字典存储ab清单文件中所有的ab包，key=crc，value=单个游戏对象的item类）

**方法：**

1. LoadAssetBundleConfig（初始化方法）：如果是编辑器下，直接返回；把热更新下载的清单列表中所有ab包存入字典

## 三、ObjectManager