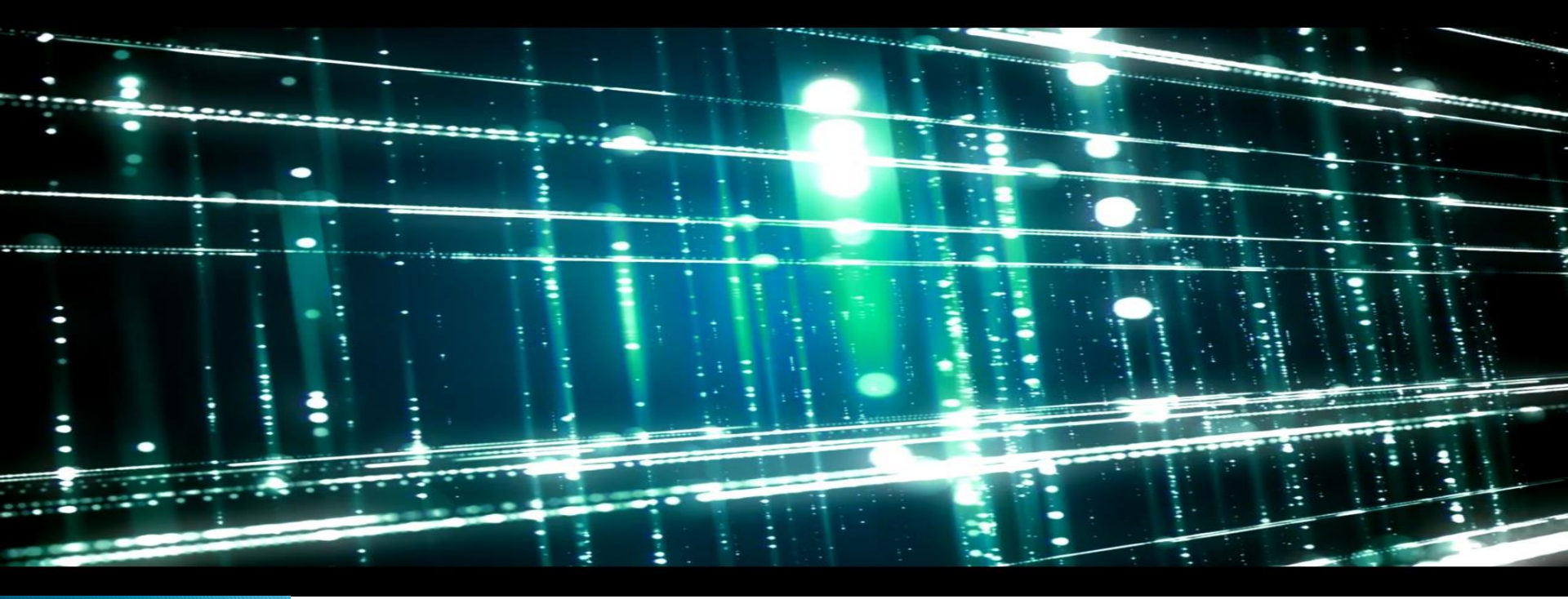


AssetBundle理论讲解

《AssetBundle框架设计》　讲师:刘国柱

AssetBundle理论讲解

- 据国外AppAnnie统计2016年中国iOS手游市场正式超越美国成为了全球第一，中国也成为了当之无愧的全球第一大手游市场。
- 目前随着国内“王者荣耀”、“阴阳师”、“球球大作战”等明星级手游的崛起，这种强交互性、注重社交性设计的的网络游戏更为年轻人喜爱。那么游戏中的更新技术便成为游戏研发人员所必备的知识技能。
- 本视频课程结合Unity引擎的AssetBundle技术向学员展示游戏更新的基本原理、开发过程、应用技巧等。

AssetBundle理论讲解

- Unity引擎的AssetBundle本质就是一种资源管理的技术，通过动态的加载与卸载资源，极大的节约游戏所占空间，而且这种技术也实现了游戏发布后关于资源的后续更新与完善，所以这也是一种游戏的实时热更新技术。

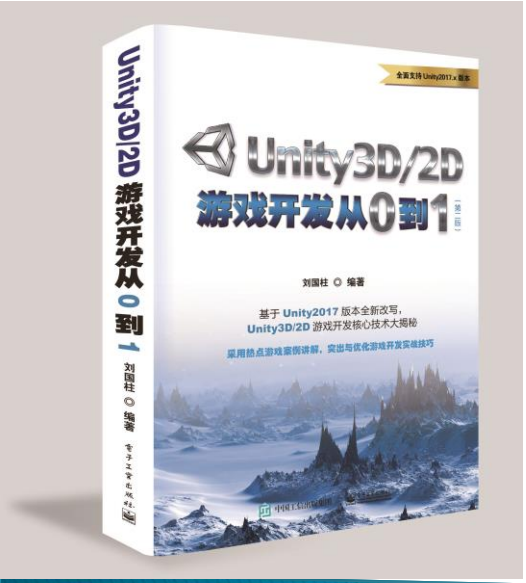
- AssetBundle是一个压缩包。 它包含模型、贴图、预制体(Prefab)、音频等资源，可以在游戏运行期被加载。

- (Unity5.x之后)AssetBundle自身保存着互相的依赖关系。压缩包可以使用LZMA和LZ4压缩算法，减少包大小，更快的进行网络传输。具体到商业游戏中，可以把游戏后期运行的大量资源都放在AssetBundle里面，可以大幅减少安装包尺寸。

AssetBundle理论讲解

- 总体来说AssetBundles可以分为以下四个部分：
- 第1：　创建AsetBundles。
- 第2：　上传资源服务器端。
- 第3：　下载AssetBundles资源。
- 第4：　加载与卸载AssetBundles资源。

前导课程说明_就业课程系列



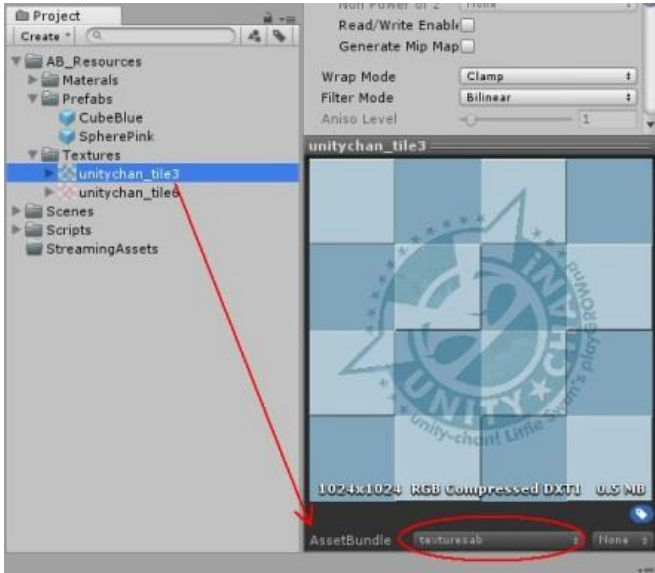
就业课程系列

前导课程说明_商业开发系列

目录

创建AssetBundle

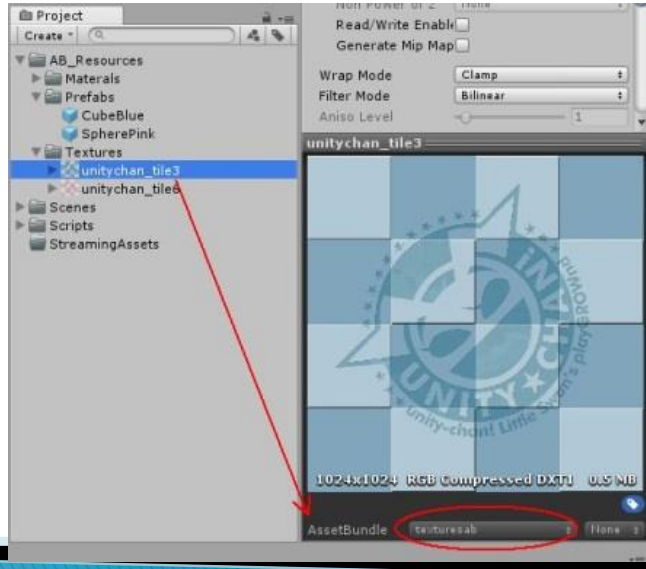
- 早期基于Unity4.x版本期间，创建AssetBundle是一件需要写大量代码且容易出错的事情,但到了Unity5.x以后这一过程已经大大简化。



创建AssetBundle

- 基于Unity2017版本创建AssetBundle可以分为以下3大步骤：

- 1： 首先定位需要打包与加载的资源，资源可以是任意类型（如：贴图、材质、音频、预设等）。在项目视图中点击资源，属性窗口下方中可以看到资源预览。 在AssetBundle后面输入需要打包的“AssetBundle名称”。



创建AssetBundle

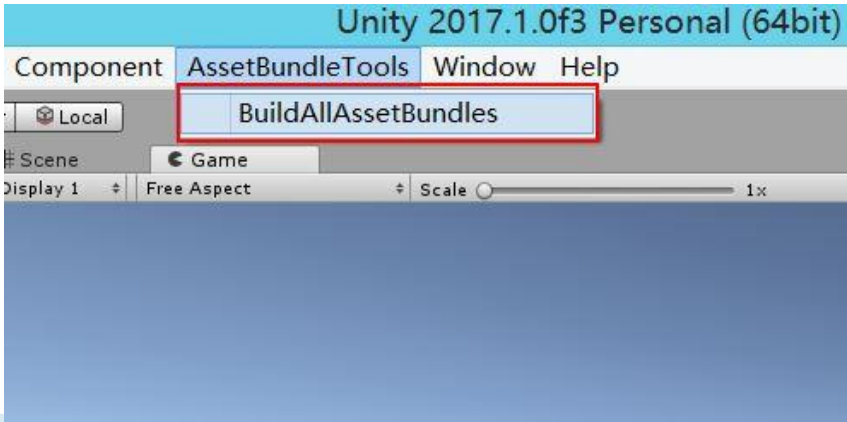
- 2： 现在编写打包脚本(BuildAssetBundle.cs)，在编写前首先确认脚本定义在“Editor”的特殊文件夹下。

- 3： 打包核心API:

BuildPipeline.BuildAssetBundles(“AB输出路径”,BuildAssetBundleOptions.None,BuildTarget.StandaloneWindows64);

创建AssetBundle

- 4:　编写脚本，在Unity编辑器顶部菜单会出现自定义的AB菜单。点击菜单后开始打包，大约几秒后在项目视图的StreamingAssets目录下我们可以看到打好包的文件资源。



目录

下载AssetBundle

- Unity目前提供了两种通过WWW类下载AssetBunde文件的方式方法。

- 第1种是“缓存机制”。采用这种机制下载的AssetBundle文件会存入Unity引擎的缓存区，通过WWW类的静态方法LoadFromCacheOrDownload实现下载。

- 第2种是“非缓存机制”。采用这种机制下载的AssetBundle文件不会存入Unity引擎的缓存区。 (www.assetBundle)

下载AssetBundle

- 通过WWW类的实例方法www.assetBundle实现下载。

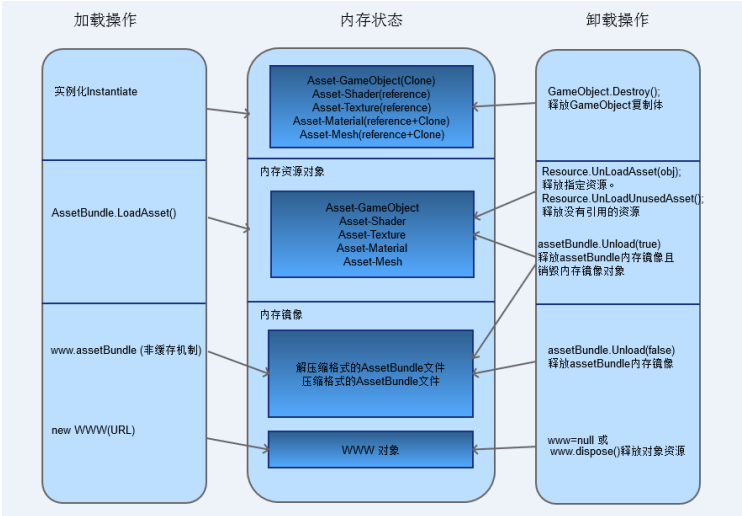
```
66    IEnumerator LoadPrefabsFromAB(string ABURL,string assetaName="",Transform showPos=null)
67    {
68        //参数检查
69        if (string.IsNullOrEmpty(ABURL))
70            Debug.LogError(GetType()+ "/LoadPrefabsFromAB()/ 输入参数 'AssetBundle URL' 为空，请检查！");
71        using (WWW www=new WWW(ABURL)){
72            yield return www;
73            AssetBundle ab = www.assetBundle;
74            if (ab!=null){
75                if (assetaName == ""){
76                    //实例化主资源
77                    if (showPos!=null){
78                        //确定显示方位
79                        GameObject tmpClonePrefabs=(GameObject)Instantiate(ab.mainAsset);
80                        tmpClonePrefabs.transform.position = showPos.transform.position;
81                    }
82                    else {
83                        Instantiate(ab.mainAsset);
84                    }
85                }
86                else {
87                    //实例化指定资源
88                    if (showPos != null){
89                        //确定显示方位
90                        GameObject tmpClonePrefabs = (GameObject)Instantiate(ab.LoadAsset(assetaName));
91                        tmpClonePrefabs.transform.position = showPos.transform.position;
92                    }
```

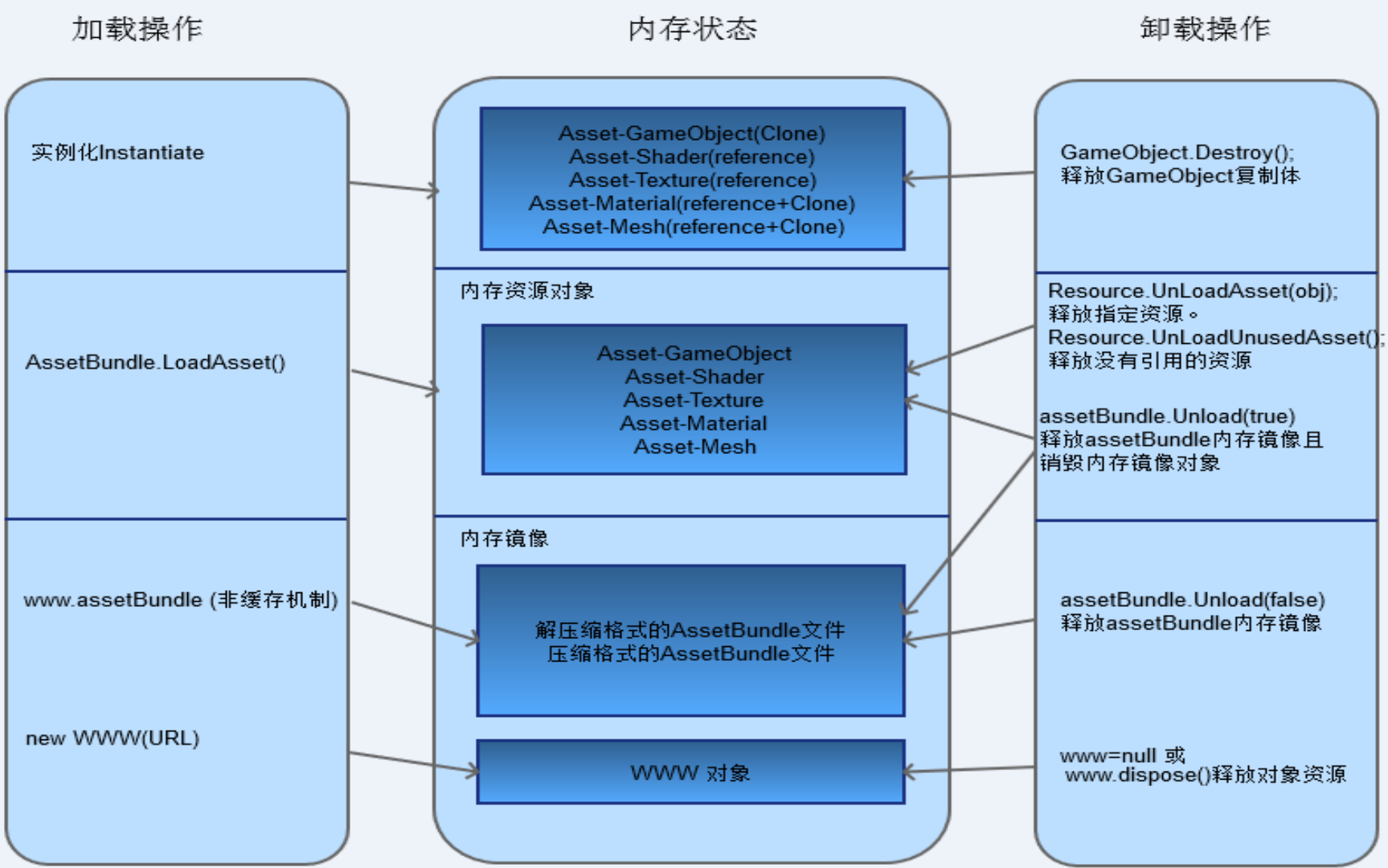
目录

AssetBundle原理讲解

- 在应用AssetBundle资源前，AssetBundle首先需要通过WWW下载到本地，然后AssetBundle在Unity引擎帮助下自动解压缩，这一过程也成为“内存镜像”。

- 然后需要加载AssetBundle到内存区域中通过相关操作，最终创建具体的游戏对象才能显示与应用。




加载操作
内存状态
卸载操作
实例化Instantiate
AssetBundle.LoadAsset()
www.assetBundle (非缓存机制)
new WWW(URL)
Asset-GameObject(Clone)
Asset-Shader(reference)
Asset-Texture(reference)
Asset-Material(reference+Clone)
Asset-Mesh(reference+Clone)
内存资源对象
Asset-GameObject
Asset-Shader
Asset-Texture
Asset-Material
Asset-Mesh
内存镜像
解压缩格式的AssetBundle文件
压缩格式的AssetBundle文件
WWW 对象
GameObject.Destroy();
释放GameObject复制体
Resource.UnLoadAsset(obj);
释放指定资源。
Resource.UnLoadUnusedAsset();
释放没有引用的资源
assetBundle.Unload(true)
释放assetBundle内存镜像且
销毁内存镜像对象
assetBundle.Unload(false)
释放assetBundle内存镜像
www=null 或
www.dispose()释放对象资源

AssetBundle原理讲解

- 值得一提的是Unity2017提供了三种不同的方法来加载已经下载的数据资源。

- assetBundle.LoadAsset();
  通过指定assetBundle包名称加载资源。

- assetBundle.LoadAssetAsync();
  异步加载模式。与上述类似，但是加载过程不会同时阻碍主线程的运行，这特别适合需要读取大尺寸资源，以及一次性读取多个资源的场合。

- assetBundle.LoadAllAssets();
  加载assetBundle中包含的所有资源对象。

目录

AssetBundle依赖关系

- Untiy4.x 之前老版本的AssetBundle系统需要自己写很多代码(BuildPipeline)，从而增加了很多学习成本。正确处理好资源的依赖关系从而保证资源完整而又不会产生重复资源，是一件比较麻烦的事情。

- 从Unity5.x开始之后的新的AssetBundle大大简化了这一操作。Unity打包的时候会自动处理依赖关系，并生成一个*.manifest文件，这个文件描述了assetbundle包大小、CRC验证、包之间的依赖关系等。

AssetBundle依赖关系

- Untiy的依赖关系处理并不是万能的，对一些复杂处理还是需要研发人员手工处理。

```
ManifestFileVersion: 0
CRC: 3019130723
Hashes:
  AssetFileHash:
    serializedVersion: 2
    Hash: d41d4fbf1144d41790fdc25265f73d44
  TypeTreeHash:
    serializedVersion: 2
    Hash: 6dcaad0a13ca8ec8df27970c98d4c899
HashAppended: 0
ClassTypes:
- Class: 1
  Script: {instanceID: 0}
- Class: 4
  Script: {instanceID: 0}
- Class: 21
  Script: {instanceID: 0}
- Class: 23
  Script: {instanceID: 0}
- Class: 28
  Script: {instanceID: 0}
- Class: 33
  Script: {instanceID: 0}
- Class: 43
  Script: {instanceID: 0}
- Class: 48
  Script: {instanceID: 0}
- Class: 135
  Script: {instanceID: 0}
Assets:
- Assets/AB_Resources/Prefabs/SpherePink.prefab
Dependencies: []
```


谢谢大家！
刘老师讲Unity系列教程