**生成assetbundle资源包**

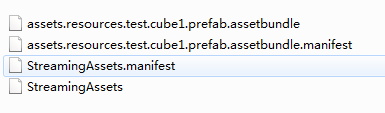
AssetBundleManifest BuildAssetBundles(string outputPath, BuildAssetBundleOptions assetBundleOptions, BuildTarget targetPlatform);

　　调用该函数，unity会自动根据资源的标签进行打包，而且是**增量打包**，  
　　a.对于资源没有变更的bundle包，不会触发重新打包；  
　　b.资源没变，即使生成目录下的bundle包被删除了，unity也不会重新打包；  
　　c.生成目录下的bundle包对应的manifase被删了，会重新打包；  
　　d.可以使用BuildAssetBundleOptions.ForceRebuildAssetBundle参数触发强制重新打包。

AssetBundleManifest BuildAssetBundles(string outputPath, AssetBundleBuild[] builds, BuildAssetBundleOptions assetBundleOptions, BuildTarget targetPlatform);

　　该函数提供了不依赖资源标签、通过代纯代码的方式打包生成资源的能力。

**生成的bundle包资源目录解析**：  
　　下图是针对Cube.prefab生成的资源目录结构：



　　StreamingAssets：一个AssetBundle包，内含AssetBundleManifest类型的Asset，记录了所有bundle包及相互间的依赖关系。  
　　　　运行时需要首先加载这个AssetBundleManifest，然后根据其提供的depence信息加载依赖的bundle包。  
　　StreamingAssets.manifest：**全局manifest**，全局manifest的名字和打包生成的目录同名，不是固定的，这里是生成在StreamingAssets目录下。  
　　Cube.assetbundle：资源bundle。  
　　Cube.assetbundle.manifest：dle一一对应，只是用来做增量build，运行时根本不需要。

**压缩格式**  
　　**(1)LZMA**：默认压缩格式，压缩比大，省空间，使用前需要解压整个压缩包；  
　　**(2)LZ4**：5.3版本新增， 40%–60% 的压缩率，开启BuildAssetBundleOptions.ChunkBasedCompression打包参数即可。  
　　　　LZ4算法是“基于块”的，因此当对象从一个LZ4压缩包加载时，仅用于该对象的相应块会被解压，不需要解压整个包。  
　　　　所以LZ4和不压缩资源一样，都可以不解压，而直接读取包中的资源的。  
　　**(3)UnCompress**：不压缩，访问最快，费空间。  
　　一组测试数据：

|  |  |  |
| --- | --- | --- |
| LZMA：1.45M | LZ4：2M | UnCompress：4M |

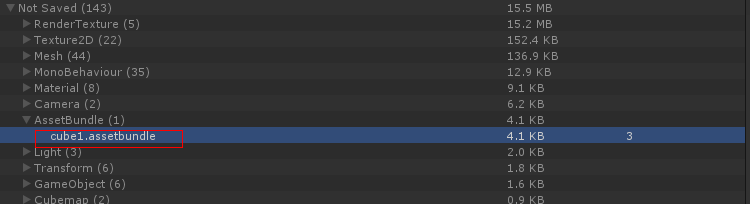
**压缩格式**  
　　**(1)LZMA**：默认压缩格式，压缩比大，省空间，使用前需要解压整个压缩包；  
　　**(2)LZ4**：5.3版本新增， 40%–60% 的压缩率，开启BuildAssetBundleOptions.ChunkBasedCompression打包参数即可。  
　　　　LZ4算法是“基于块”的，因此当对象从一个LZ4压缩包加载时，仅用于该对象的相应块会被解压，不需要解压整个包。  
　　　　所以LZ4和不压缩资源一样，都可以不解压，而直接读取包中的资源的。  
　　**(3)UnCompress**：不压缩，访问最快，费空间。  
　　一组测试数据：

|  |  |  |
| --- | --- | --- |
| LZMA：1.45M | LZ4：2M | UnCompress：4M |

使用方式建议：  
　　(1)随包资源StreamingAssets:  
　　　　未压缩或LZ4压缩：LoadFromFile；  
　　　　LZMA压缩：可使用 WWW.LoadFromCacheOrDownload解压缩到本地磁盘。  
　　(2)热更新资源：LZMA+WWW.LoadFromCacheOrDownload+Caching.compressionEnabled；  
　　(3)加密资源:LZ4+LoadFromMemory；  
　　(4)自己压缩的资源：UncompressAssetBundle的AssetBundle包+自己的算法压缩+LoadFromFileAsync。

对于shader一定要注意重复打包，要么将使用的shader配置在Always Included Shaders，要么将shader也单独打包

**(2)内存分析：**  
　　一个AssetBundle压缩包并不大，所以同时缓存100个不释放也没多大（4.x这个东西还是很大的，不能缓存太多）。  
　　AssetBundle创建，会在Not Saved/AssetBundle分组下多出一个bundle，大小4.1k：



　　LoadAsset(cube1)后，会在Assets下多处cube1产生的asset：

