**Asset Bundle Internal Structure**

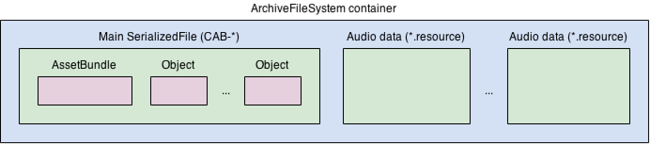
**Asset Bundle的内部结构**

An AssetBundle is essentially a set of objects grouped together into a serialized file. It is deployed as a data file which has a slightly different structure depending on whether it is a normal bundle or a scene bundle.

AssetBundle本质上就是一个由对象组成的序列化文件。他被部署成一个数据文件，普通包和场景包在结构上会有一些略微的不同。

**Normal AssetBundle structure.**

**普通AssetBundle结构**

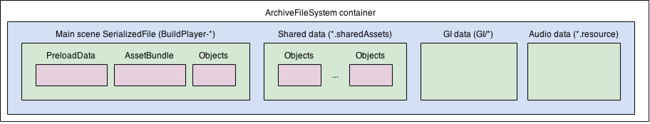


The internal structure of a Normal Asset Bundle

普通Asset Bundle的内部结构

**Streamed scene AssetBundle structure.**

**流场景AssetBundle结构**



The internal structure of a Streamed Scene Asset Bundle

流场景AssetBundle的内部结构

Both types are archived and compressed in a same way.

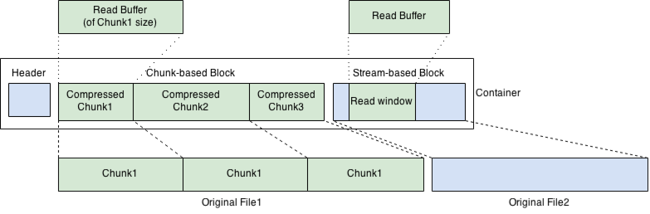
两种类型都以同样的方式归档和压缩。

**AssetBundle compression**

**AssetBundle 压缩**

The ArchiveFileSystem container has the following structure:

ArchiveFileSystem容器的结构如下



The ArchiveFileSystem container structure

ArchiveFileSystem容器结构

The Compressed blocks shown above might have chunk-based compression or stream-based compression. Chunk-based compression (LZ4) means that the original data is split to chunks (subblocks) of equal size and that chunks are compressed independently. You should use this if you want realtime decompression - random read overhead is small. Stream-based compression (LZMA) uses the same dictionary when processing the whole block, it provides the highest possible compression ratio but supports only sequential reads.

上面显示的压缩后的块可能是基于堆压缩的或者基于流压缩的。基于堆的压缩（LZ4）意思是原始的数据被划分为相等大小的多个堆（子块），每个堆的压缩都是独立进行的。你应该使用这种（基于堆的压缩）如果你希望实时解压缩的话 - 随机读取的开销是比较小的。基于流的压缩（LZMA）在处理整个块时使用的是同一个字典，他提供了最高的压缩比但是只支持连续读。