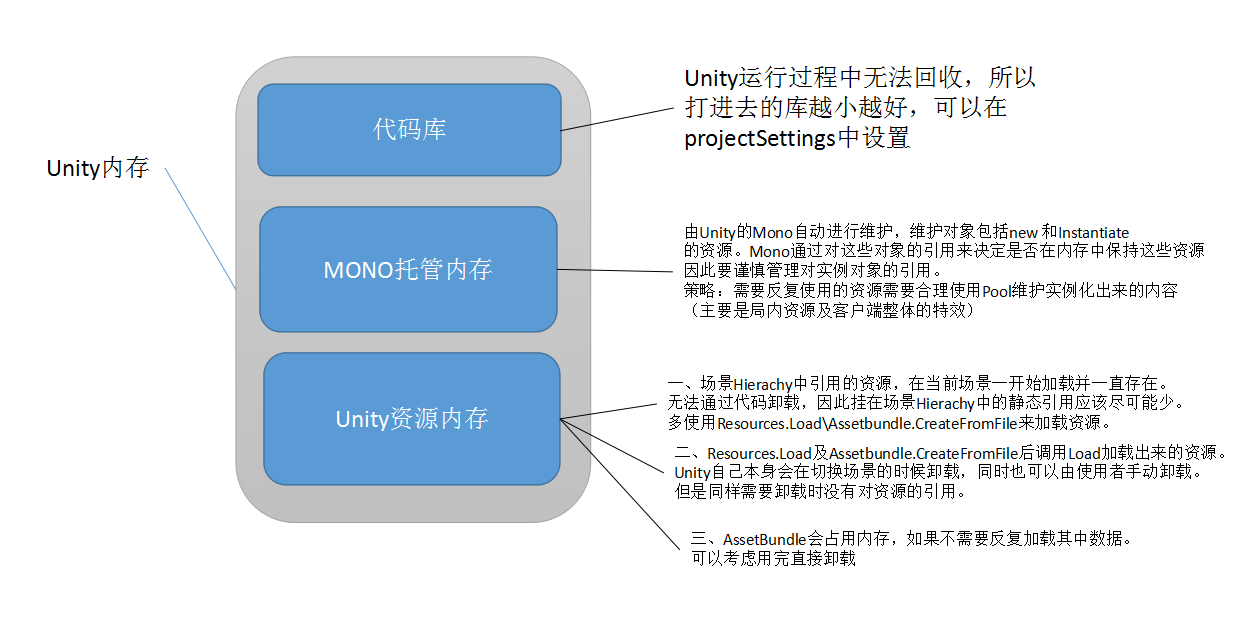
DGM客户端内存资源管理方案

|  |  |  |  |
| --- | --- | --- | --- |
| **版本** | **更新时间** | **更新内容** | **更新人** |
| **V0.1** | 2016-03-09 | 建立文档 | 张青 |

**需求**

合理管理内存中资源的生命周期，保证尽可能小的内存开销，并同时保证尽可能稳定的CPU性能开销。

**Unity内存的分配与管理原理**



Unity的自动内存管理机制：

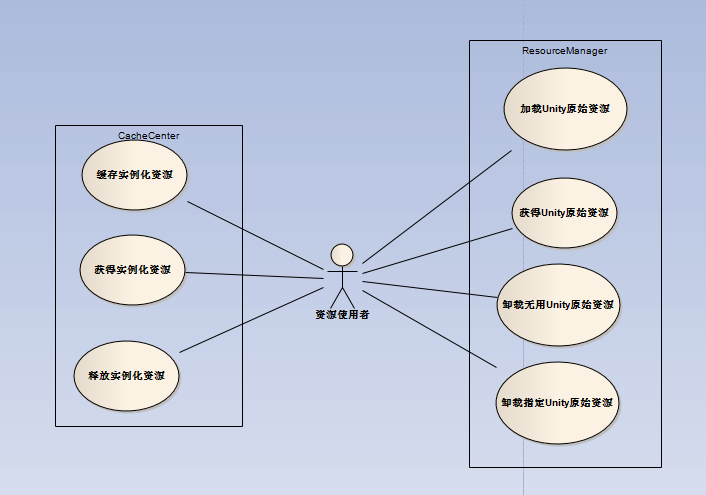
1. Mono会对托管内存中通过代码new 出来的对象及使用GameObject.Instantiate实例化的对象进行管理，在Unity场景切换的时候自动进行回收。这两种对象我们称为**实例对象**。当**实例对象**不再被引用被引用、超出生命周期的时候，Mono会自动对这些已经没有用的**实例对象**进行回收，释放占用的内存。因此对代码中的**实例对象**的引用及生命周期进行管理，可以较好管理Mono托管内存的大小。通过Instantiate生成的**实例对象**可以通过Destroy手动释放。
2. Unity会对通过Resources.Load及Assetbundle.Load加载出的资源进行管理，在Unity场景切换的时候自动进行回收，这类对象我们称为**Unity原始资源**。并且这类资源可以通过Resources.UnloadunusedAssets及GC.Collect可以手动进行回收。（调用GC.Collect只是建议底层进行回收，极端情况有可能被延后执行。）只有没有被引用的对象才会被回收。
3. AssetBundle会占用一定的内存，在通过Assetbundle.Load加载出其中的资源之后就没有用了。只需要缓存住加载出来的资源，就可以通过Assetbundle.Unload(false)卸载掉Assetbundle。
4. 在.unity场景中直接引用的资源（包括这些直接引用资源依赖的资源）我们称为**静态资源**，在场景加载时载入内存，并且**不能通过代码释放**。只有当场景切换的时候才会被释放。

**总体方案**

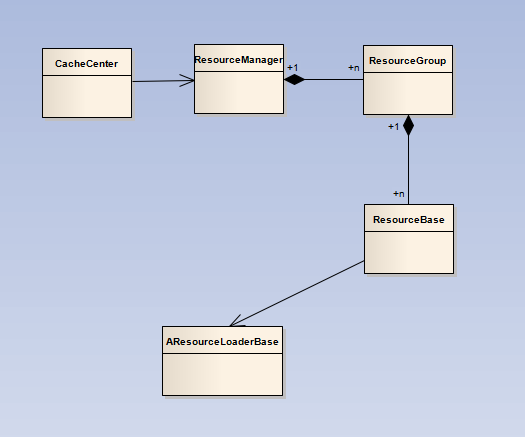
内存的管理从这四部分入手：

1. 尽可能少的使用**静态资源**，使用动态加载的方式使用与管理资源。这部分由客户端框架在UI切换流程与进对局流程中进行设计与规范。
2. AssetBundle在加载完其中的**Unity原始资源**后，调用AssetBundle.Unload(false)将Assetbundle从内存中卸载。这部分由资源管理器自动处理。
3. Resources.Load及AssetBundle.Load加载出的**Unity原始资源**，在使用后立刻将引用设为null。这部分由资源管理器自动处理。但在需要时再使用Resources.UnloadunusedAsset清除资源。
4. 频繁Instantiate\Destroy的资源，我们称为**实例化资源**，大量频繁使用的实例化资源使用对象池来管理。

**资源使用用例**



领域设计



ResourceManager:资源管理器，资源获取的入口。所有动态加载的资源都从这里得到。

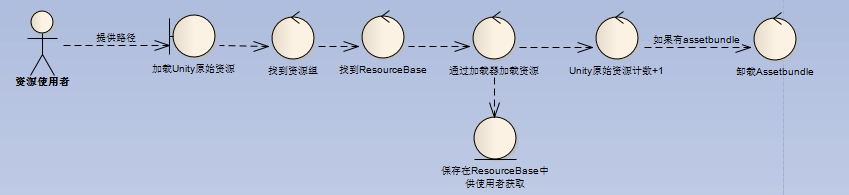
ResourcesGroup：资源组，每一类资源使用一个资源组进行管理。每一个资源组定义了资源的类型、资源用途、资源加载方式、资源依赖关系等，并根据这些不同实现不同的策略。

ResourcesBase:资源的基类，定义了资源的引用、释放及资源的计数管理。

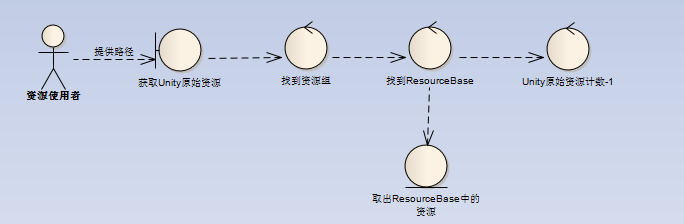
AResourceLoaderBase：资源加载器。一类加载器实现一种资源加载方式，可以在不同的使用环境下（编辑器、ios、android）使用不同的加载策略（从resources中加载，从assetbundle中加载，从音频、文本文件中加载等）。

CacheCenter:实例化对象池，对于频繁使用的资源，使用实例化对象池先进行缓存。需要用时再从缓存池中获得。

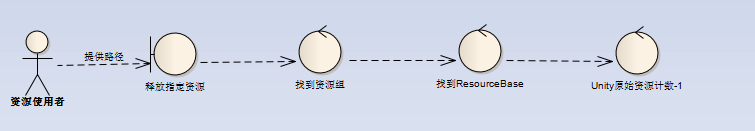
**加载Unity原始资源流程**



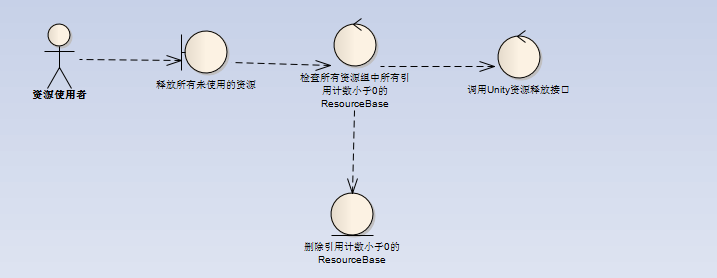
获得Unity原始资源流程



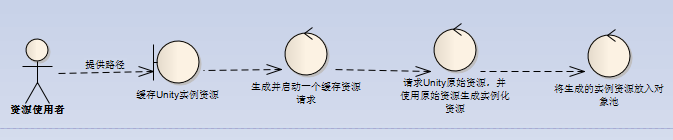
释放指定Unity原始资源（减少指定资源的引用计数，并不直接释放）



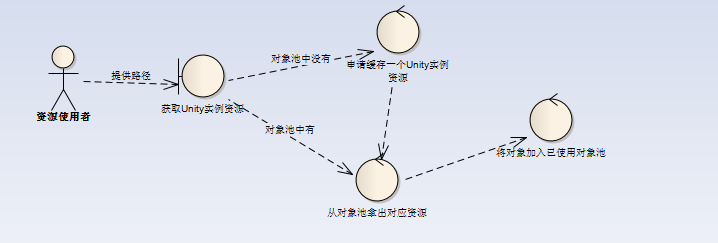
释放全被未引用的原始资源



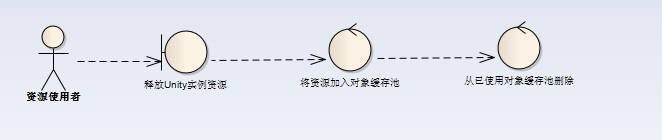
缓存实例化资源



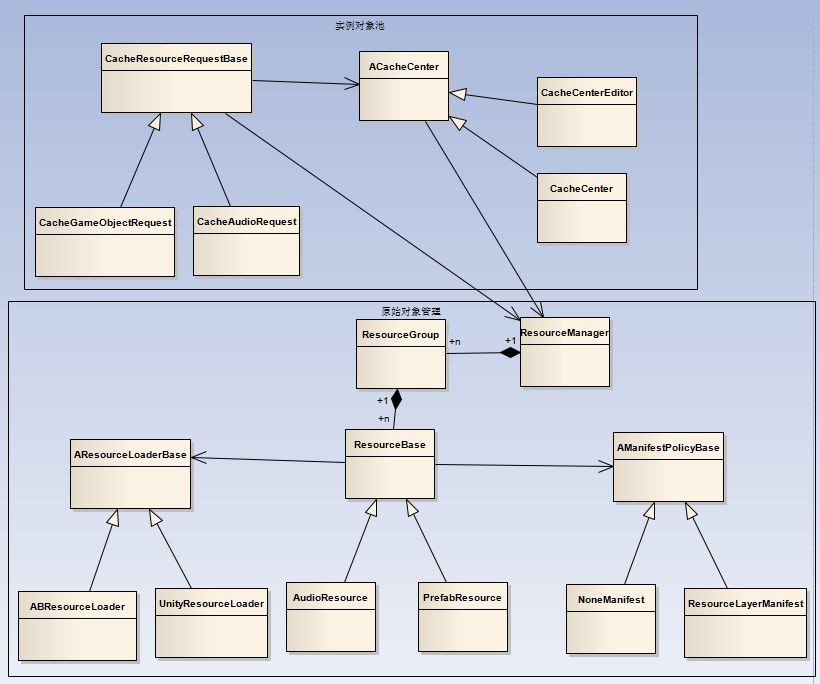
获得实例化资源



释放实例化资源



概要类设计



其他

* 局内所有实例资源需要使用对象池预先缓存使用。
* 局外需要在切换界面时调用释放资源接口。释放上一个界面已经不用的资源。
* 资源使用者没有特殊需求的情况下不能将从资源管理器中拿到的Unity原始资源赋给一个成员变量、静态变量或者全局变量