# **一文读懂使用图集，降低drawcall**

作者：[vian](http://gad.qq.com/user/index?id=55567)

链接：<http://gad.qq.com/article/detail/286187>

**实际操作测试，一文读懂使用图集，降低drawcall**

1.首先我们来了解一下Batch。

批渲染（Batch）

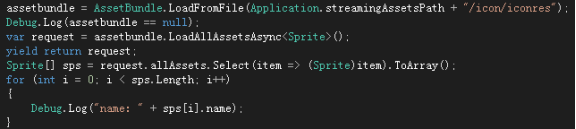
batch render 是大部分引擎提高渲染效率的方法，基本原理就是通过将一些渲染状态一致的物体合成一个大物体，一次提交给gpu进行绘制，如果不batch的话，就要提交给很多次，这可以显著的节省drawcall，实际上这主要节省了cpu的时间，cpu从提交多次到提交一次，对gpu来说也不用多次切换渲染状态。当然能batch的前提一定是渲染状态一致的一组物体。

2.Batch相关测试

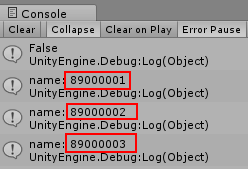
一，图集测试。我们将物品图标生成图集（可以去了解texturepacker），如下图所示：



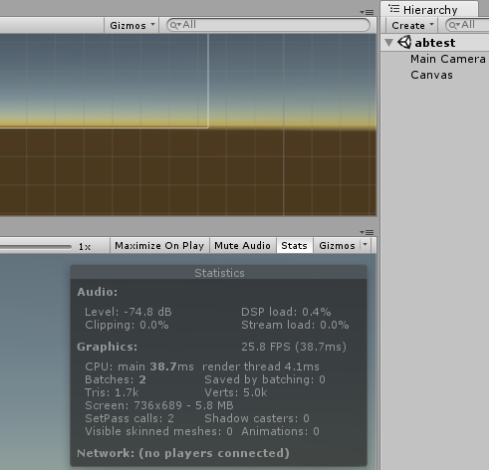
然后将icon目录下这张图集打ab包，打包方法在前面的文章里有介绍（深入AssetBundle打包加载使用）。实际应用中，我们可能有上百个道具icon，这里只是方便测试。这个不是本文重点，主要是测试查看batch。我们将生成的ab包加载进来：



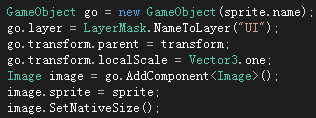
输出下数据是否正确？输出如下所示：



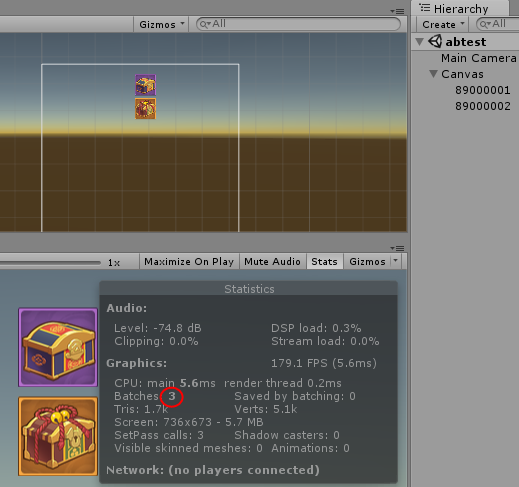
经过上面操作我们已经将整个图集数据加载出来，我们也可以通过id来获取对应道具图标了。下面我们创建显示图集里的小图标，为了好进行对比，创建前我们先看看操作前的batch数据如下图所示：



创建icon的显示，相关代码如下所示：



先显示图集里的两个小图标，在查看batch数据如下所示：

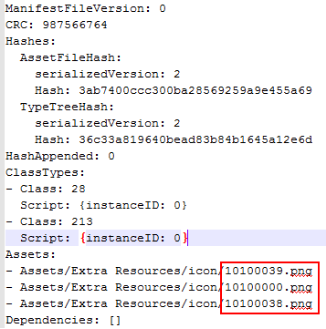


两个图标由于是在同一图集里，一次batch即可，符合性能优化需要。

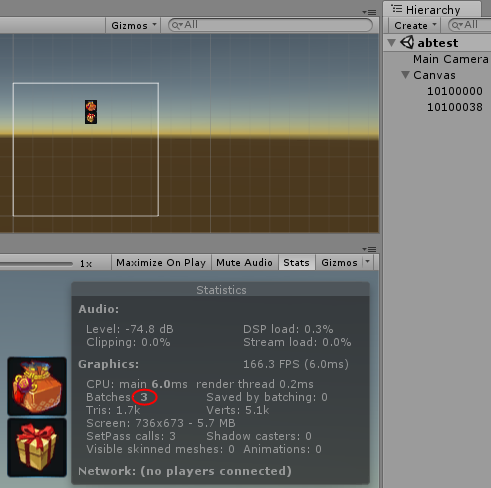
二．‘非图集测试’。当然这里说的非图集仅仅是资源目录放的是散图，实际操作后的并不是真正意义上的非图集，我们是利用打ab包的特性来自动化完成图集，这里面同样要先看前面的文章（深入AssetBundle打包加载使用）。资源目录具体如下：



同样我们需要将icon目录图标进行打ab包。成功后其描述文件如下：



加载新的ab包，也显示其中两个图标出来，具体数据如下图所示：



同样这里也是只有一次batch，符合要求。

以上，一般图集会比散图大，因为图集是2的次方, 会有一些空余。至于在实际项目中我们需要直接用图集还是用散图，看具体美术资源管理需要或项目喜好了，毕竟图集也好散图也好我们最终都会打到ab包。个人觉得用图集的方式好处就是能更好的去优化图集大小，而打ab包都是自动化的。另外在打ab前我们也可以测试一下图集大小，推荐尽量控制1024\*1024内，unity对于2的次方的素材能提供ETC压缩，压缩大小还可以。

3.注意错误的使用

直接引用目录下的图标，由于图标未在同一图集，会造成batch消耗增加，如下图所示：

