# **Unity相关优化小结**

作者：vian

链接：<http://gad.qq.com/article/detail/287053>

**以下是unity开发中需要注意的一些优化点，送上的都是比较接地气的干货！**

**1.静态美术资源的输出**

能合在一起整体输出的资源，尽量不要切开输出。



如上图分享按钮根据登录渠道qq微信展示不一样，道具展示也动态的，其他元素基本可作为一张整体背景资源输出即可。

**2.UI相关大背景图处理方式**

一般为了表现丰富，ui整张大背景资源也是常见的。



如上图ui大背景，别放到游戏内别打到ab包里去，一般可放到CDN上，用的时候根据背景的url用www加载下来。注意要美术严格控制输出资源大小！根据项目需要可定义一个限制大小如不能超过500K。

**3.图集大小测试检查**

实际开发中并不是把资源往目录放下就ok，一般一个目录下的资源我们会让其打包图集。尽量保证图集不超过1024\*1024，我们可对打图集进行测试分析,以便提前知道图集大小问题。



以上例子目录下三张资源，测试图集256\*256，符合正常。如果你在开发过程中发现测试图集大小超出了1024\*1024，那么这时候要注意，先跟美术沟通优化资源，需要时可以分目录放，其实也就是分图集来处理。为什么说分图集，跟网络流量一个概念，流量大网络容易拥塞。所以尽量避免大图集的出现。虽然同图集对DC有优化，但并不是图集越大越好。

**4.打ab包注意**

提出公用目录，将项目内公用性比较强的资源统一放到该目录，注意不要把公用性较低的资源也一起放了，跟进实际项目去决定。因为该目录打ab包内容是被依赖的，进游戏就需要先把它加载进来的，否则其他ab包加载进来找不到依赖资源也显示不正常。

实际开发中很多情况会遇到Aui内有个X图标与Bui用的一个，可能Cui也有用到，但是这几个ui并不是进游戏就能打开或有入口触发打开的。这就是为什么不把X图标放到公共目录的一个原因。这中情况我们也可提出一个目录整理存放这些被公共到的资源，但该目录不进行ab打包，而是直接打包引用该目录资源的各个预制体，该情况下资源是被打包多份的。有以内存换外存的意思。

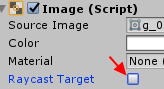
**5.UGUI一些优化**

5.1 透明边框避免过度绘制over draw，可去掉勾选Fill Center；

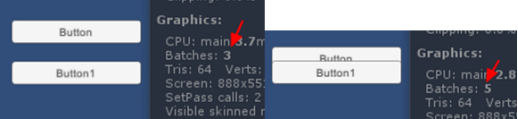


降低overdraw还有一种情况，系统Aui打开覆盖遮挡Bui，此时注意将Bui禁掉。

5.2 大部分Image和Text不需要响应事件，可以去掉RaycastTarget勾选；



5.3 布局UI时，同一深度的UI（Item）不要出现重叠。



5.4 对于有特效显示的ui，最好新挂个canvas节点供特效使用。

**6.代码一些注意**

6.1 尽量避免使用foreach；

6.2 尽量避免使用+来进行字符串连接；

6.3 尽量不要update函数中做复杂计算，如有需要，可以隔N帧(FixedUpdate)计算一次；

6.4 尽量减少函数调用栈，用x = (x > 0 ? x : -x)代替x = Mathf.Abs(x)；

6.5 使用内建的常量，例如Vector3.zero等等，避免频繁创建相同的对象；

6.6 引用一个游戏对象的逻辑，可以在start函数里去定义它，如记录transform等；

6.7 尽量不主动调用GC，养成良好代码习惯。如果需要可在游戏暂停，切场景GC一次；

6.8 避免频繁的调用Find、GetComponent等；

6.9 用go.CompareTag (“XX”)来代替go.tag == “XX”；

6.10 尽量不要动态的Instantiate和destroy object，使用object pool；

以上是简要整理分享的干货，关于unity相关优化是聊不完的话题，随个人开发习惯与经验累积，优化的道路才会越走越顺畅！

上文涉及到优化相关知识，附：

合并网格合并贴图，优化DC相关：

<http://gad.qq.com/article/detail/286354>

图集测试，优化DC相关：

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

资源管理，ab包相关：

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

Apk包生成、更新相关：

<http://gad.qq.com/article/detail/286454>