游戏前端优化

2019.11.18会议整理

# UI优化

## 重构、动静分离

为了优化网格重建（CPU）。

每次绘制都需要重绘顶点属性。Unity以Canvas为单位，每一帧都刷新Canvas下的东西。

**解决办法：**将有改变的东西（位置、颜色之类的）放到一个Canvas下。

## Prefab拆分

为了优化实例化Prefab加载耗时。

**解决办法：**将可拆分Prefab（比如Prefab内有多个页签，每个页签对应不同的东西，可将各页签内的东西）拆分成多个子Prefab,初次打开时对应子Prefab时加载一次，关闭时放入缓冲池。

## Drawcall

在Unity中，每次引擎准备数据并通知GPU的过程称为一次Draw Call。

**原理：**每一次的 Draw Call 会产生一个 Batch，而 Batch 里装的是物件顶点资料。Batch 由 CPU 通过 “驱动程式” 将顶点资料送往 GPU，GPU接手后将物件画在画面上。

**解决办法：**Batch：同一个图集里的图片可以合批，合到一起绘制。

Unity 在 Player Setting 里的两个功能选项 Static Batching 与 Dynamic Batching。Static Batching 是将标明为 Static 的静态物件，如果在使用相同材质球的条件下，UNITY 会自动帮你把这两个物件合併成一个 Batch，送往 GPU 来处理。Dynamic Batching 是在物件小于300面的条件下(不论物件是否不静态或动态)，在使用相同材质球下，UNITY就会自动帮你合併成一个 Batch 送往 GPU 来处理。

## 频繁显隐优化

**解决办法：**需要隐藏时Scale改为0或移出屏幕外。

# 其他优化

## LOD

只渲染近处的东西

## 换图。

近处的东西用高清图，远处的东西用模糊图。

# 附录（真木分享）

关于为啥要做动静分离， 这边摘录一下问答。 简单来说 绘制每帧还是都要绘制的（所以drawcall能降则降，能合批则合批）， 动静分离 主要是降低 canvas这边 对Mesh（网格）的重建开销：

Q5：动静分离或者多Canvas带来性能提升的理论基础是什么呢？如果静态部分不变动，整个Canvas就不刷新了？

在UGUI中，网格的更新或重建（为了尽可能合并UI部分的DrawCall）是以Canvas为单位的，且只在其中的UI元素发生变动（位置、颜色等）时才会进行。因此，将动态UI元素与静态UI元素分离后，可以将动态UI元素的变化所引起的网格更新或重建所涉及到的范围变小，从而降低一定的开销。而静态UI元素所在的Canvas则不会出现网格更新和重建的开销。

Q3：能否就UGUI Batch提出一些建议呢？是否有一些Batch的规则？

在 UGUI 中，Batch是以Canvas为单位的，即在同一个Canvas下的UI元素最终都会被Batch到同一个Mesh中。而在Batch前，UGUI会根据这些UI元素的材质（通常就是Atlas）以及渲染顺序进行重排，在不改变渲染结果的前提下，尽可能将相同材质的UI元素合并在同一个SubMesh中，从而把DrawCall降到最低。而Batch的操作只会在UI元素发生变化时才进行，且合成的Mesh越大，操作的耗时也就越大。

因此，我们建议尽可能把频繁变化（位置，颜色，长宽等）的UI元素从复杂的Canvas中分离出来，从而避免复杂的Canvas频繁重建。

Q6：UWA建议“尽可能将静态UI元素和频繁变化的动态UI元素分开，存放于不同的Panel下。同时，对于不同频率的动态元素也建议存放于不同的Panel中。”那么请问，如果把特效放在Panel里面，需要把特效拆到动态的里面吗？

通常特效是指粒子系统，而粒子系统的渲染和UI是独立的，仅能通过Render Order来改变两者的渲染顺序，而粒子系统的变化并不会引起UI部分的重建，因此特效的放置并没有特殊的要求。

Q5：对于UI网格更新和重建的两个情况，我有如下两个疑问想了解下：

1）我对于网格重建的理解是：如果UI的Mesh发生了变化，比如血条UI中血量变了可能导致Mesh变化，从而发生了重建。但网格更新是什么样一种情况呢？UWA Blog的原文中提到了“只在其中的UI元素发生变动（位置、颜色等）时才会进行”，这句话是否可以理解为颜色变化和位置的变化产生了网格更新？

2）是否即使UI没有变化，每帧中仍然是会产生Draw Call？这样想是因为每帧都要显示UI，而这个UI的数据即使有缓存可能也是让CPU省事了，但是GPU仍旧需要进行渲染的工作，从而CPU还是要调用Draw Call。如果是这样，那么其实UI的Mesh重建因为需要CPU进行计算所以开销比较大；而UI由于位置发生变化导致的Mesh更新，因为只是重新提交给GPU UI模型的世界坐标，因此消耗是比较小的；而UI由于颜色变化导致的Mesh更新，其实消耗就更小了，这里只是在提交渲染状态时改变颜色即可。不知道上面的理解是否正确。

答：

更新指的是UI元素本身的某些属性发生变化，从而需要重新生成，或者更新顶点属性。比如颜色变了，在UGUI中颜色的变化是通过修改顶点色实现的，所以就需要更新UI元素对应的每个顶点的顶点色属性（可以认为就是修改下某个数组里的数值）。位置移动一般是不会造成顶点属性的变化的。所以总的来说，“网格更新”更新的是顶点属性。

UI元素和别的网格不同点在于，UI的网格是需要进行合并的，并且在UGUI中是以Canvas为单位的，在提交GPU之前，同一Canvas下的所有UI元素都会被合入一个Mesh中（但包含多个SubMesh）。所以位置的移动，顶点属性的变化，都会导致这个Mesh要重新合并，也就是网格重建。这也是为什么说要“动静分离”的原因，完全静态的Canvas是不需要重建的，但只要里面有一个UI元素在动，就会引起Canvas的重建。

1、打开一个全屏UI， 如果发现， 这个UI的名字一直在上面， 说明这个UI上有个元素 在动， 需要优化一下（比如把一直动的 加个canvas （一般是加 GraphicRaycaster组件，不加这个点击会有问题））。

（有的只是简单滚动不会重建的； 有的改了内容一般都会重建）

2、打开一个全屏UI，如果发现 Canvas（Canvas）这个根节点出现了， 说明 前面一个UI被暂时关闭时， 动画效果没暂定， 还在那边刷。（ 现在暂时关闭UI是 不SetActive(false)的， 是关Canvas的，所以可能脚本还在跑）

所以一个优化原则： UI在OnDisable时，必须要把一些动态变化的元素绘制停下来。