NGUI中关于UIPanel.LateUpdate中GC Alloc的优化

注意：所有本次优化相关的代码都用OPTIMISE\_NGUI\_GC\_ALLOC宏区分开了，使用的NGUI版本是3.11.4。

# UIGeometry的verts、uvs、cols、mRtpVerts缓冲池

UIGeometry记载了UIWidget的顶点信息，可以采用缓存池的策略，当UIWidget.OnFill的时候确定了需要顶点数量的情况下从缓存池申请，当UIWidget.OnDestroy的时候放回缓存池，回收的时候以顶点数量从小到大插入，申请的时候找到第一个满足需求顶点数量的。

缓冲池的实现



UIWidget.OnDestroy的时候放回缓存池

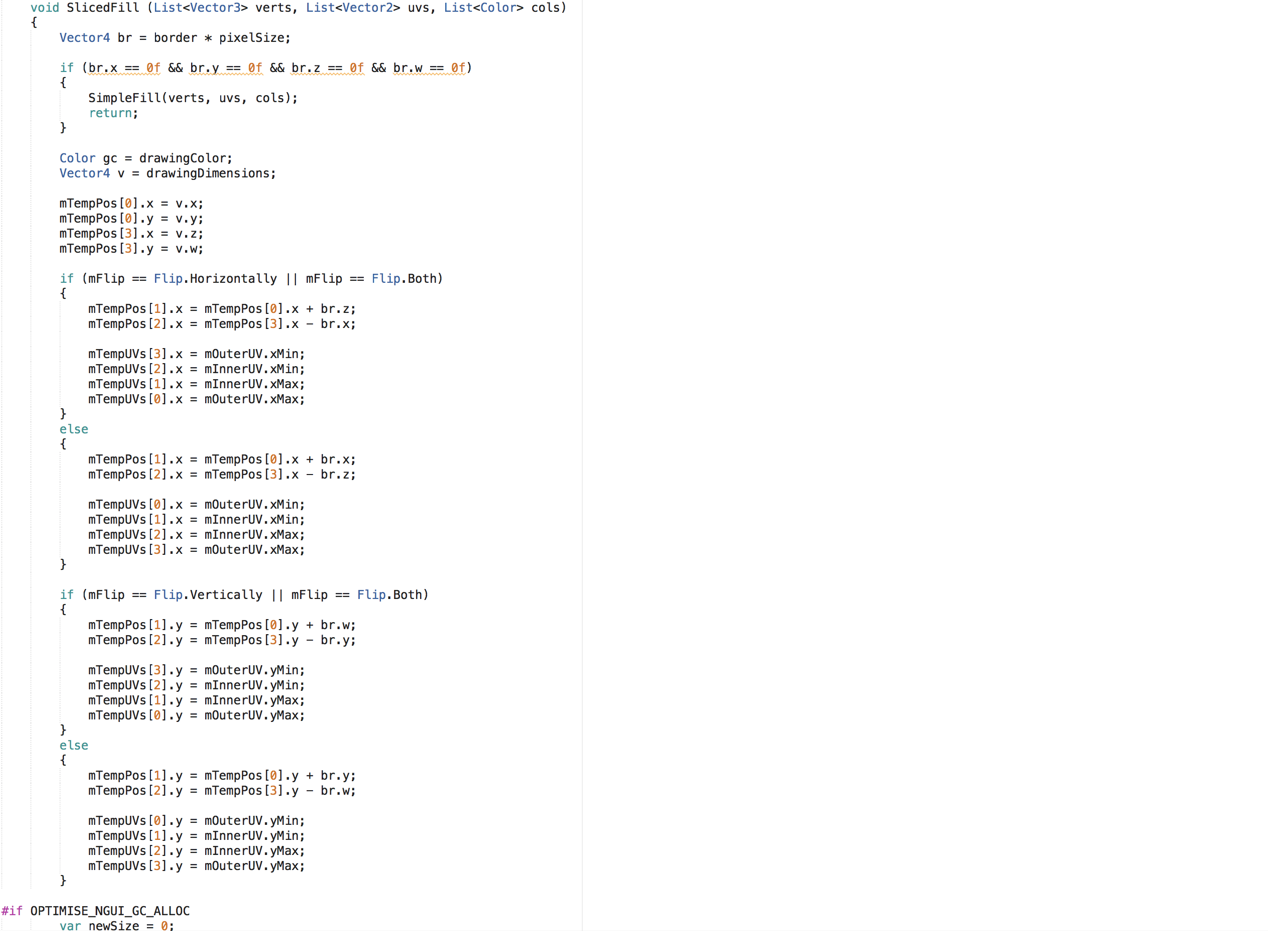


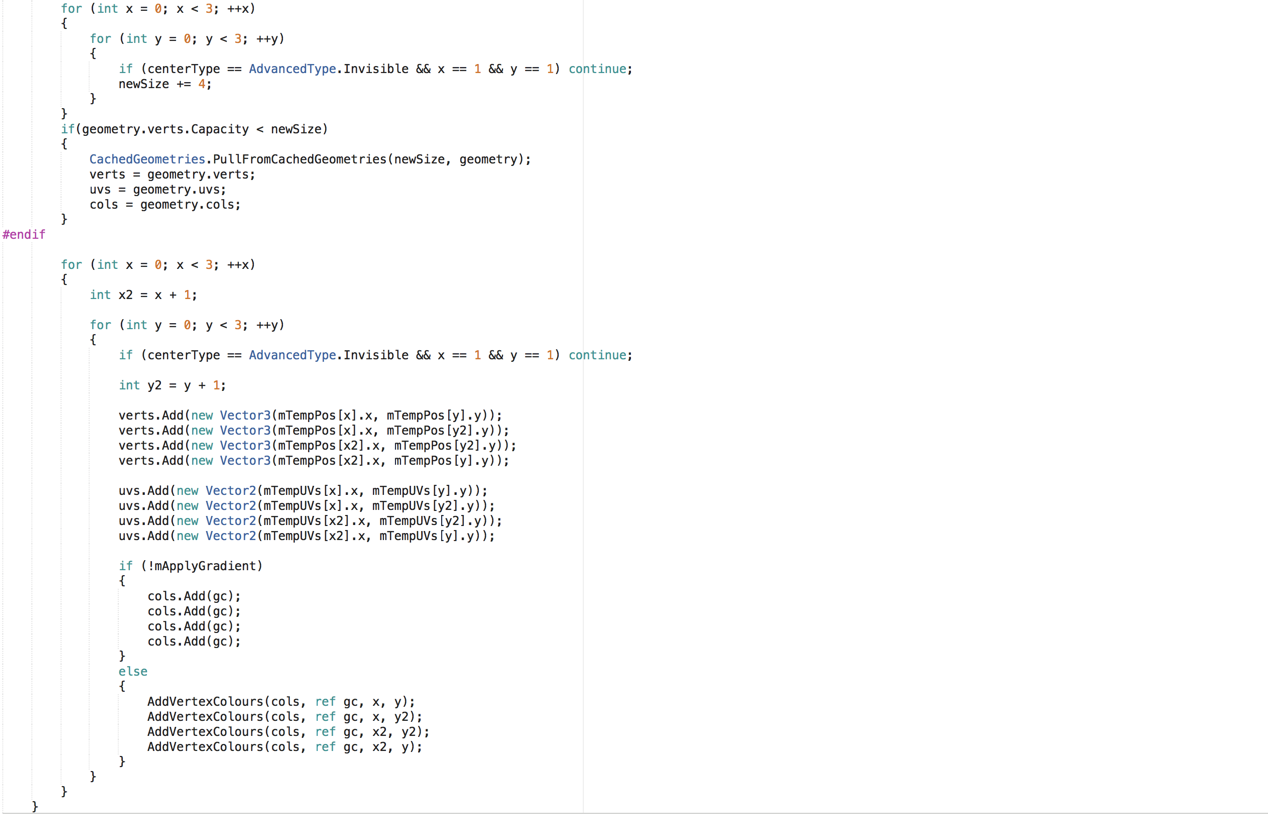
UISprite和UITexture我优化了三种类型Simple和Sliced和Tiled，剩下两种留给读者实现吧☺

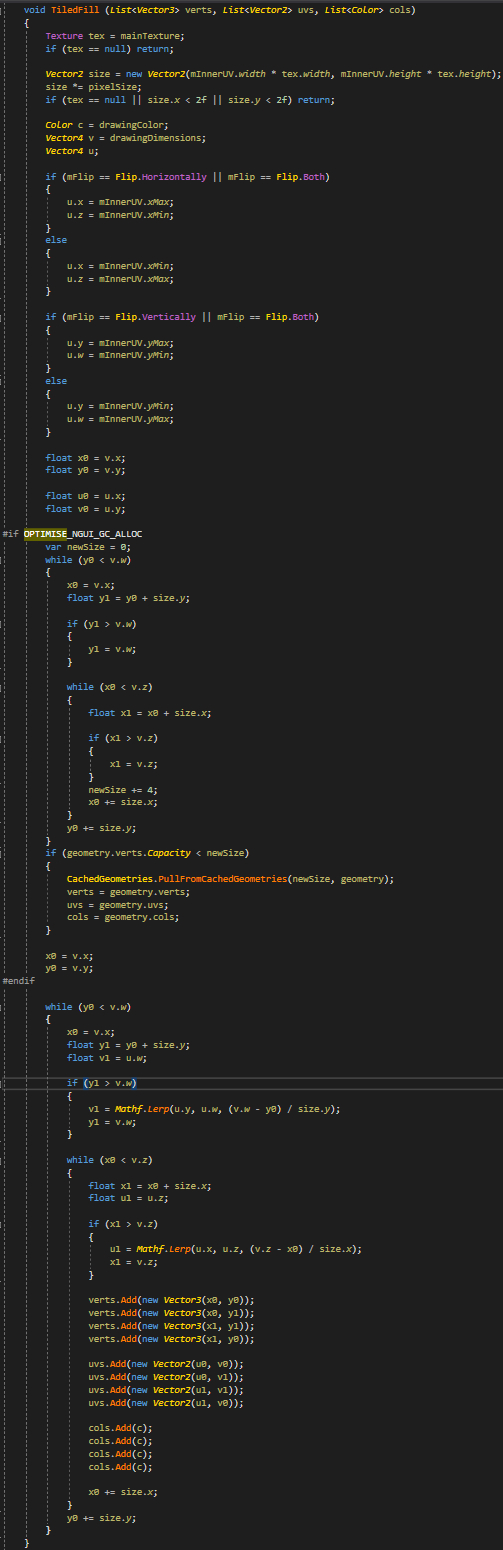
对了UI2DSprite我也没有优化，所以在NGUI里用了Unity的Sprite的同学自己实现吧☺

以下代码出现在UIBasicSprite.cs里，共三个函数





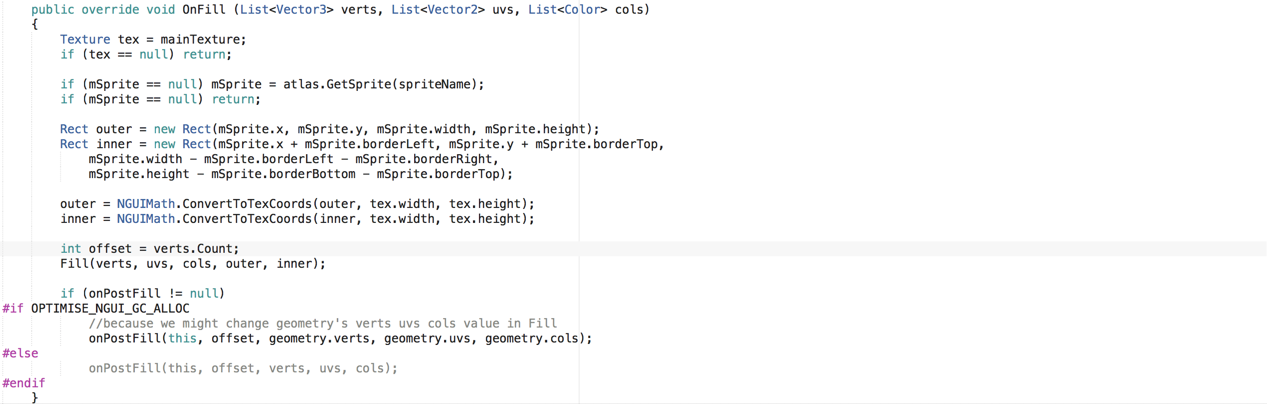




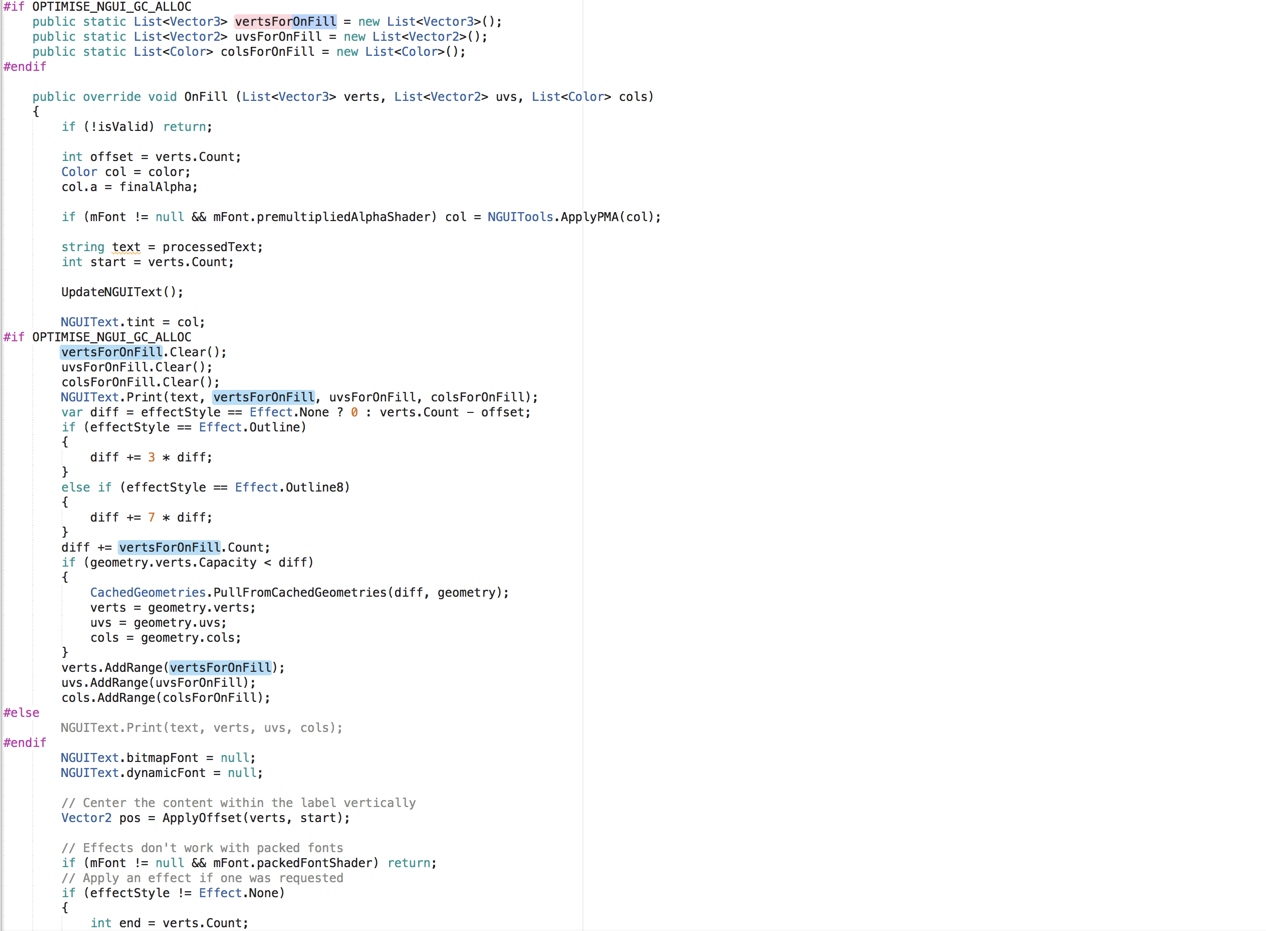
因为上三个函数都可能在UIBasicSprite.Fill中给UIGeometry的几个List赋予新的实例，所以在UITexture 的OnFill中相关代码需要改成这样



UISprite的OnFill中相关代码需要改成这样



UILabel.OnFill里额外提供了一个缓冲供NGUIText.Print使用，里面也考虑了UILabel带有描边和阴影时的优化

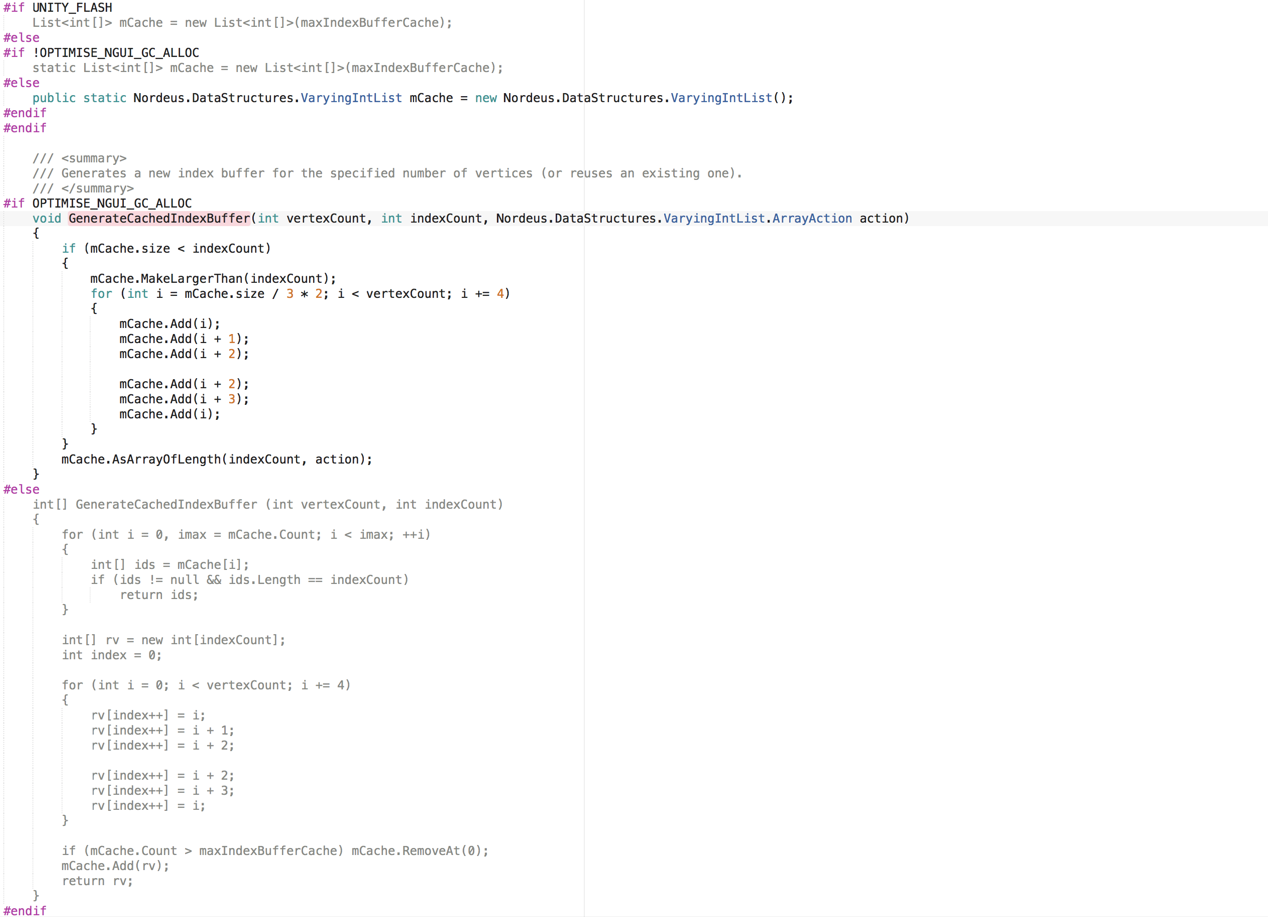




# NGUI创建mesh时顶点索引缓冲的优化

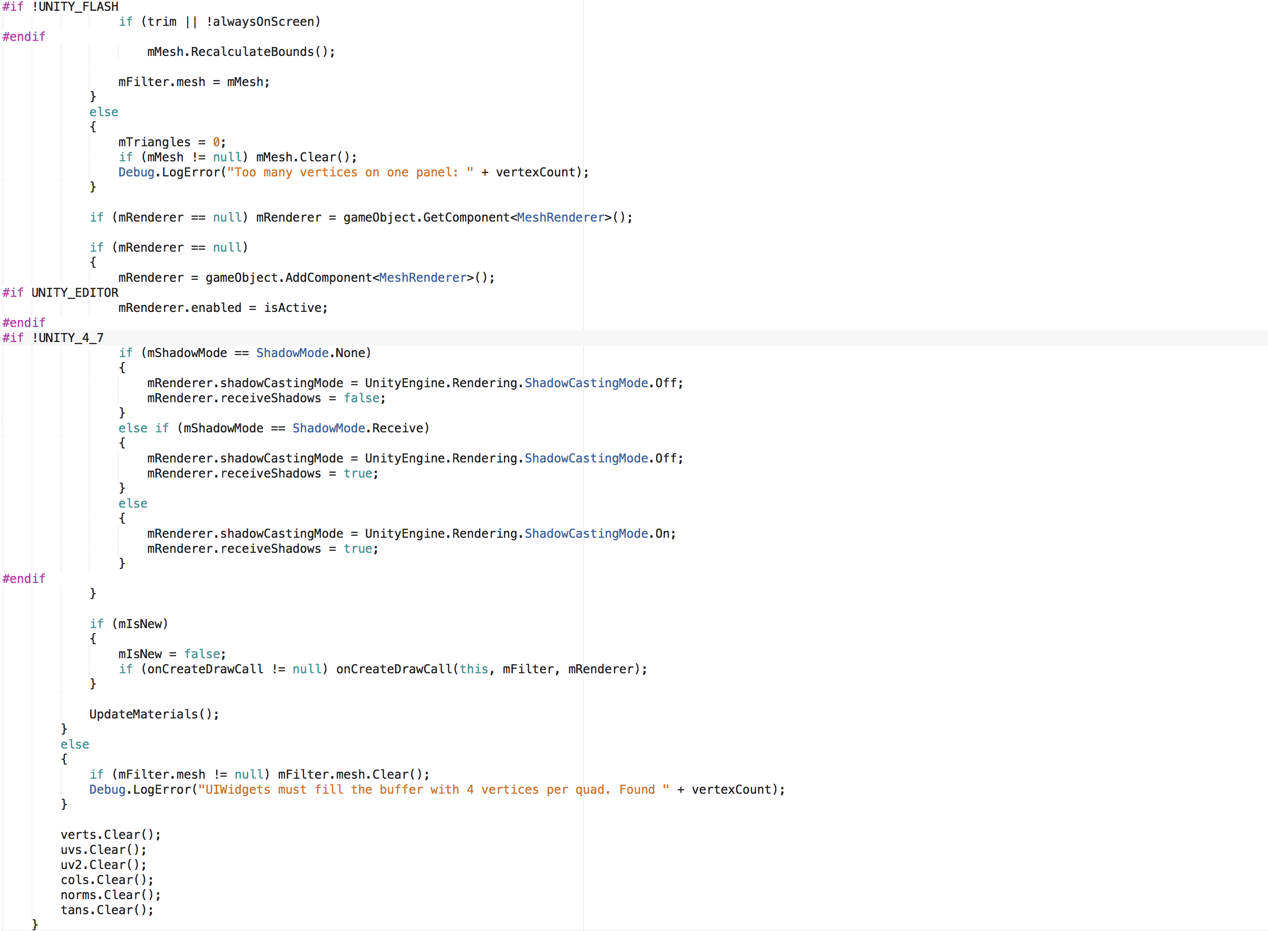
从原来的顶点索引的生成算法来看，假设有一个顶点索引缓冲长度为9000，另一个有1w，那么这两个缓冲的前9000个索引的值是一样的，而且Mesh.triangles需要顶点索引缓冲的数量为顶点缓冲数量的三倍，所以最好是有个能够能动态调整Array.Length但Array的元素可以不发生变化的办法，正好Nordeus在Unite2015上分享了一个VaryingList（<https://github.com/Nordeus/Unite2015/tree/master/VaryingList>，把BufferedList.cs和VaryingList.cs下载到工程，开启unsafe就行了）可以做到，它主要是通过unsafe的方法来实现我们要的功能。这样，我们只需要一个顶点索引缓冲，而不是原来的10个，而且我们只在缓冲不够大的时候从新分配内存并只填充增加了的顶点索引缓冲，而当需要的顶点索引缓冲比当前的小的时候只需要通过unsafe方法设置下长度就行了

代码在UIDrawCall.cs





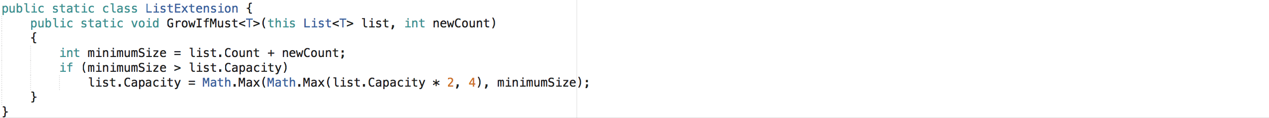




# 插入前充分设置List<T>.capacity来减少GC Alloc

没错，就是这样一个基本的操作。

先让我们来包装一下这个操作吧，List<T>内部用GrowIfNeeded（相信同学们在deep profile里看过他很多次了吧）来进行扩容，我不想用反射，所以直接把代码拿过来用扩展做



在UIPanel中，需要对FillAllDrawCalls和FillDrawCall进行优化，收集每个UIDrawCall对应的UIWidget和总的顶点数，然后一次性对UIDrawCall里的几个List<T>一次性设置capacity

