合批规则 ： 简单来讲就是先计算层级号，

再计算材质，再计算纹理；

相叠的两张图片中，会优先渲染被叠在下面的那张，

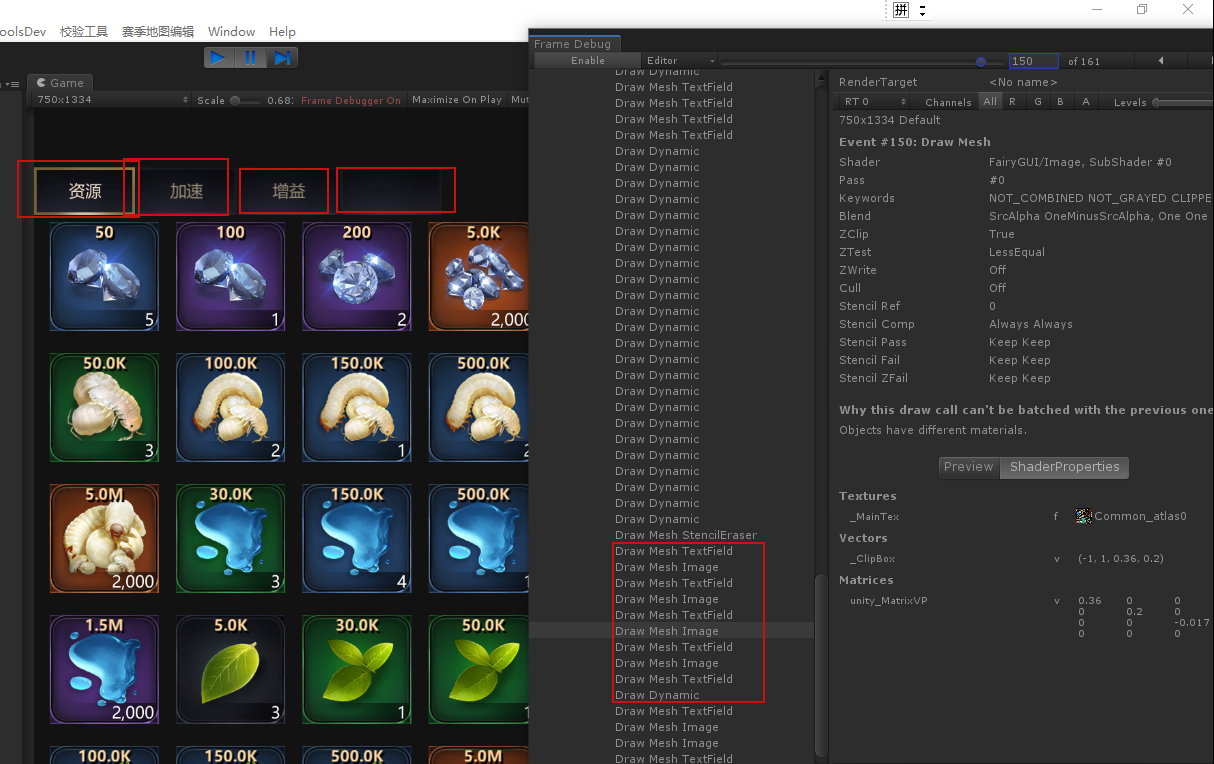
然后在渲染上面那张，如果两个图片的材质，

纹理相同那么他们同为一个层级（就会被合批为一个drawcall），

否则都是不同层级（从0开始加1）

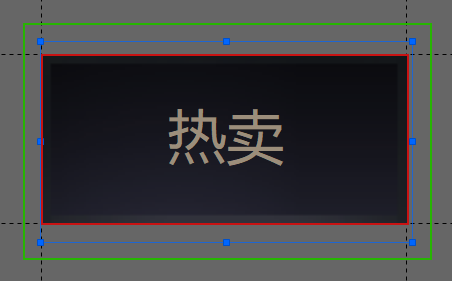
情景1： 有时候会看见前后两个drawcall都是文本，

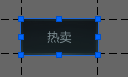
为什么不能合批呢？就是因为他们层级号不一样



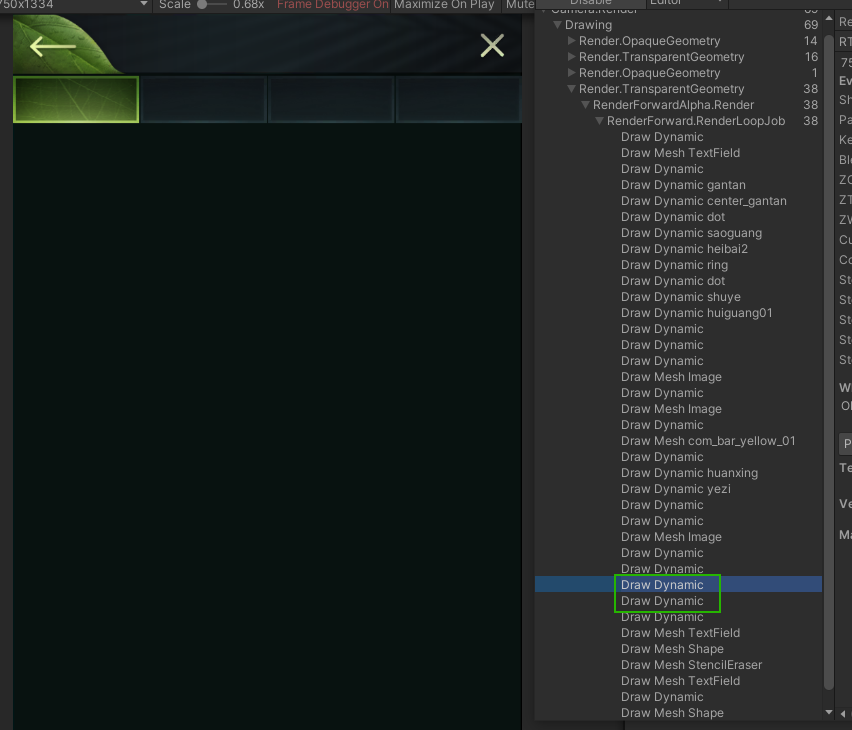
情景1案例：背包主界面上面的页签按钮，里面文本超出组件范围，导致渲染是几个页签文本层级不一样不能合批，

解决办法：将文本调整至小于等于组件大小（这样就从4个按钮，8个drawcall降至2个drawcall）

修改前：蓝色的是选中的text

修改后：

修改后这里只用了2个drawcall并且和顶部合批（一个底图一个文本）



打断合批原因：目前遇见最多的就是材质不一样

会打断合批的fgui原件：空loader， 图形，文本（文本比较特殊，虽然会打断但层级管理得当可以在一个drawcall里面全部绘制），

遮罩（本来可以理解为图形，但是用途不一般我就分开了）

项目中常见打断合批的obj：特效

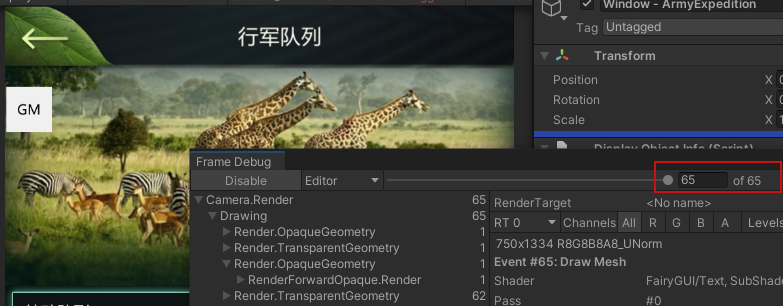
关于遮罩的特别说明：列表为例 ： 如果列表有遮罩那么列表的drawcall单独算，

如果列表内组件有遮罩，那么列表的组件组件间的合批将被打断

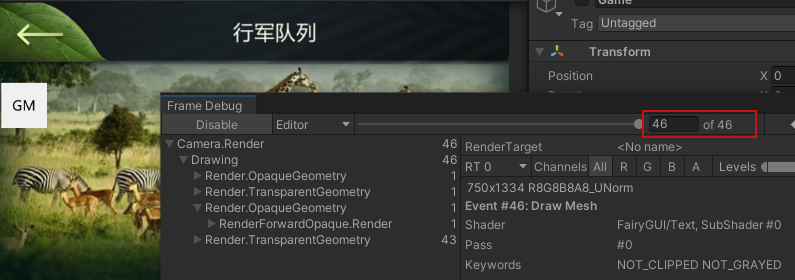
关于特效的特别说明：列表为例 ： 和遮罩一样，如果列表内组件有特效，

那么列表的组件组件间的合批将被打断【这个地方有个案例就是行军队列界面，列表在组件有特效和无特效的drawcall有很大的不同】

有特效drawcall ：

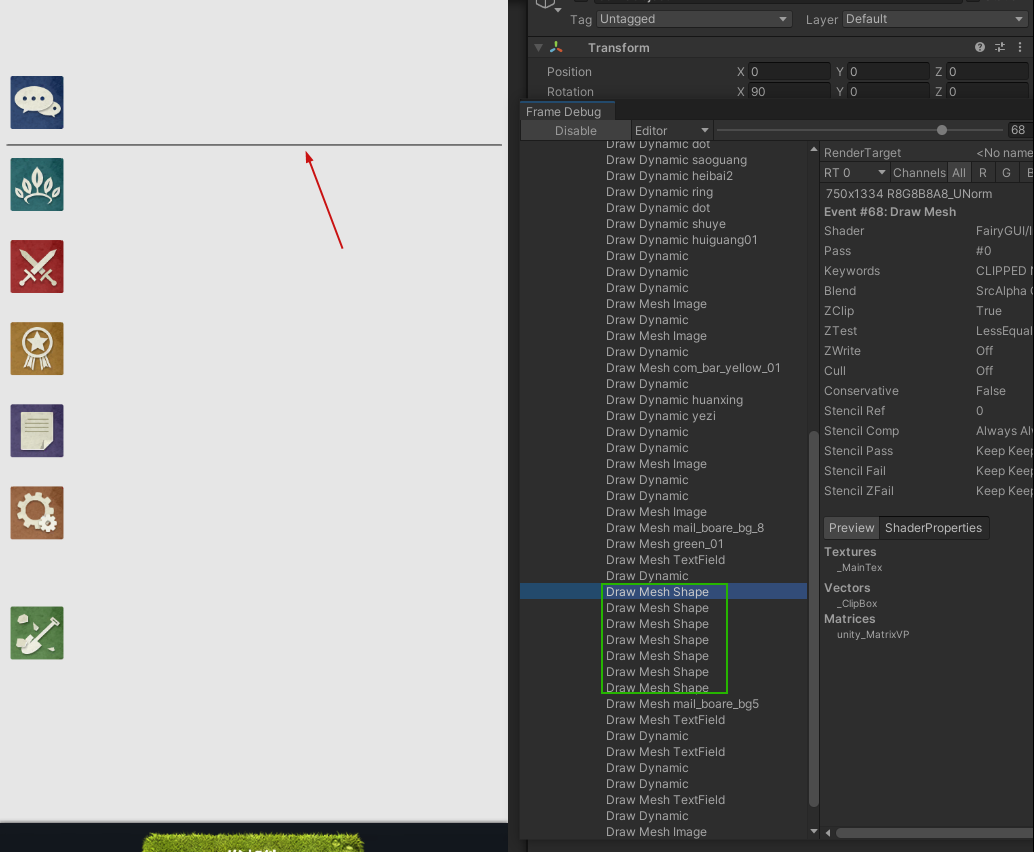


无特效drawcall：



关于图形特别说明 ： 我们图形通用用来装载obj、当地板、分割线等，

建议我们得背景、分割线不要用图形这样必然打断合批，如果分割线出现在列表中，那么drawcall爆炸；例子是邮件主界面（下图绿色框起来的全是分割线）



减少合批记录：1、组件最底层不能为空loader，检测点击的loader,可以放一种离loader最近的图片进去然后把透明度设置为0

2、组件内的图片尽量来着同一个包，如果有多个，把相同包内资源位置尽可能放一起

3、建议组件中原件的顺序可以为loader/图片 - 文本

（文本和图片/loader 不要交叉，一般交叉不会出现问题，但是如果交叉的图片出现在不同的图集那么这个时候会打断合批，文本批次增加）

4、组建中套用组件，这个时候比较复杂，因为很多通用组件中原件多，

这样层级数也不一样，那么我们把原件放在自己底层原件能够合批的位置，特效节点直接扔顶层

1. 然后就是复制的文本合批，当组件过于复杂的时候，我们要注意文本的边界问题，尽可能的不要压在来源于两个包的图片上。
2. 有时候为了文本能够合批，可以给文本增加底图（透明度为0）把层级统一。
3. 所谓层级其实不是fgui的顺序， 可以理解为视觉看上去的层级【这里有个注意的地方，有边缘空白的图片，空白处是不会加入层级计算的】
4. 组件内原件尽量不要超出组件范围，不然层级不好控制
5. 代码中使用变灰会打断合批

批注：以上例子中的界面都来源于103项目

学习链接：https://blog.csdn.net/wanttokonw/article/details/118060632?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%2522165847410916782391826150%2522%252C%2522scm%2522%253A%252220140713.130102334.pc%255Fall.%2522%257D&request\_id=165847410916782391826150&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~first\_rank\_ecpm\_v1~pc\_rank\_34-4-118060632-null-null.142^v33^pc\_rank\_34,185^v2^control&utm\_term=fgui%20drawcall&spm=1018.2226.3001.4187

https://www.fairygui.com/docs/unity/drawcall