这周主要遇到的问题是，新加的高斯模糊在项目中造成闪退问题。没必现问题，但是因为出现几率比较大，通过多次重现后，观察日志发现，在绘制屏幕的时候，因为渲染队列未清空，造成大量的unknow comands in renderqueue(未知的渲染队列)。因为对引擎底层渲染流程不太理解，尝试通过另一种方式去实现高斯模糊。

本身的思路是：使用rendertexture绘制一次屏幕，拿纹理创建精灵再去进行模糊，加入父节点后再一次绘制屏幕，最后将绘制的加到layer中去，第一次的精灵就直接删除，因为如果不再一次绘制屏幕，会因为精灵一直在进行模糊处理导致帧率下降的非常厉害。

尝试修改的思路为：使用自带的截屏功能进行截屏->保存截图->创建精灵->模糊处理->再截图->保存截图->删除精灵->使用模糊图片创建精灵。使用截屏来直接代替rendertexture来达到模糊的效果，但是使用发现，截屏造成的卡顿十分明显，二次截屏造成的卡顿就达到不可接受的程度，尝试后只能抛弃这种思路。

最后，只能尝试去网上查阅资料来解决第一个做法存在的问题。

<http://discuss.cocos2d-x.org/t/bugs-for-unknown-commands-in-renderer/16865>经查阅之后发现，属于引擎本身存在的问题，因为使用的cocos版本没有解决这个存在的问题，所以只能通过修改源码来改进这个问题。

遇到问题之后，比较困难的阶段属于定位问题，因为这一次在测试提出的闪退的现象，大多都是处于打开二次弹窗之后，而这些二次弹窗在以前都是没问题的，通过查看手机保存的日志排除了释放导致的问题之后，剩下的最大可能性就是新加的高斯模糊导致的。在具体定位到问题所在的时候，因为非必现的出现频率导致使用插件来断点定位问题比较困难，所以采取了观察运行时出现的日志来找到具体的闪退问题。

在解决问题的时候，我一般采取都是这两个方式来辅助，断点或者直接重现看日志；还有一种比较少用的就是直接用手机连接ecplise重现看日志（这种方式一般我都是用在接sdk的时候解决问题用的）。常用两种方式的价值，断点来观察问题所在价值还会比查看日志比较有效率，因为很多时候查看日志都不能直接看到直接的报错日志，而是利用自己强行加入的输出方式在运行时观察，这样耗时比较多。

1)、上周在解决一个短线重连弹窗还存在的问题时候，意识到很多自定义弹窗没有父类，管理起来比较麻烦。

2)、标签页优化后，在接受消息的时候要先判断是否初始化或者可见性，遇到一个报错是收到协议后发个消息叫panel刷新界面，而那个panel刚好是标签也，而且是没打开过的标签页，直接就报错\_skin是空值，是因为标签页在优化之后，在打开的时候不会渲染所有的uitab下面的panel，所以在接受消息的时候要先做判断

存在的问题，刚打开模块却刚好被释放，现在还是会在打开module的时候遇到\_view是空值的情况，在正常情况下\_view都是有值，只有在释放的时候遍历了self置空