用玄学线来分析应用是否流畅的总结

抛开应用的内容是否吸引（新闻类）应用是否必须（12306），

单纯决定一个应用好不好，能不能留住用户，应用的流畅度应该是一个非常重要的因素。

给用户在视觉上，直观的看来就是这个应用卡不卡，一个应用上滑下拉左划右划界面半天半天才有反应，

说得好听一点，叫这个应用可优化空间很大，难听一点 就是这个应用真垃圾。

当然了，应用的卡顿，由很多原因造成，例如：

1. 网络慢；
2. 手机后台应用太多，ROM不够用；
3. 在主线程中进行了很多复杂运算；
4. View过于复杂，GPU处理跟不上；

列举的前两种原因开发人员没法直接解决，第三种原因是开发人员是智障，这里着重研究第四种。

===================================================================

写在前面（有很多连接，首先得搞清楚玄学线用来干嘛）

优化view的话，我们借助开发者工具里的Monitor- Profile GPU Rendering



吐槽一下，某度搜索玄学线基本就是那几个淘宝例子文章反复抄，而且还是基于安卓5.0，稍微有点用的这个

<http://blog.csdn.net/xu_fu/article/details/45008779>

今天我们用6.0版本的玄学线，

很明显的看出，6.0线比5.0的颜色多了，这也是说明可监测的指标页更多了。

更加直观。



1. 什么是玄学线？

用于在屏幕上实时显示GPU渲染每一帧图像花费的时间（单位：ms）组成柱状图。

1. 玄学线的含义

分析5.0的

<http://blog.csdn.net/xu_fu/article/details/45008779>

关于6.0的Stackflow上有个相关提问：

<http://stackoverflow.com/questions/33451381/colors-of-profile-gpu-bars-on-android-m>

当然，这是目前我找到资料里最全的

<http://lilei.work/2016/02/25/Android-Performance-Patterns-s5ep10-Profile-GPU-Rendering/>

油管上还有个

<https://www.youtube.com/watch?v=erGJw8WDV74&list=PLWz5rJ2EKKc9CBxr3BVjPTPoDPLdPIFCE&index=10>

自带字幕分析。

先补完入门基础，然后打开调试，你会发现一个很坑的事情，

是文档/回答给的那些颜色，和手机屏幕显示出来的有色差，肉眼分辨起来很累。

乃至于开发者官方文档给的IDE调试图也是色差严重

<https://developer.android.google.cn/studio/profile/am-gpu.html>

强行没关系，还是得用。

1. 使用玄学线分析问题

本来我想拿澎湃新闻来做模板进行对比的，后来在研究过程中发现…澎湃的编程方式…文章最后我详细讲。

所以这里我直接拿自己手上在做的app做分析

如图所示（从上至下）

1. Process glSwapBuffers：代表着CPU通知GPU“你已经完成视图渲染了”，不过在这里CPU会等待GPU的回话，当GPU说“好的知道了”，才算完事儿。假如橙色部分很高的话，说明当前GPU过于忙碌，有很多命令需要去处理--（参考<http://mobile.zol.com.cn/566/5661167.html> ）；
2. Execute Display List ：代表了“执行时间”，它指的是Android渲染引擎执行图形驱动层中这些绘制命令的时间，假如当前界面的视图越多，绘制时间越长（与长度正比）；
3. Sync&Upload：这项指数衡量了bitmap被同步到GPU的耗时，越大的图像，数值越高，通常处理高像素的图片时会导致飙升。

减少同时展示的图片数量，或者对图片进行预处理，降低图片尺寸可以有效降低数值。

1. Update Display List：代表视图绘制所花费的时间，表示视图在界面发生变化（更新）的用时情况，越长说明当前视图越复杂；

实际上，在这里1234所耗费的时间是可被接受范围内，并不存在卡顿的情况。

1. Measure/Layout：过高的指数，表明页面布局过于复杂（层级太深），也可能是由于双重布局消耗了帧频（Double Layout Taxation）。

（有关Taxation参考：<https://www.youtube.com/watch?v=dB3_vgS-Uqo> ）

此处我的布局是一个赛事Fragment里包含三个（积分，射手，赛程）Fragment，业务要求点击底部数据Tab对该赛事进行数据刷新请求并更新view，

整个布局比较简单，但是嵌了两个RelativeLayout（供选择显示那个子模块），而更新数据列表时要根据当前选择的模块进行对应数据的清空，赋值，更新，而其他两个子模块的数据不变化，这里耗时长的原因可能是由于触发了Double Layout Taxation。

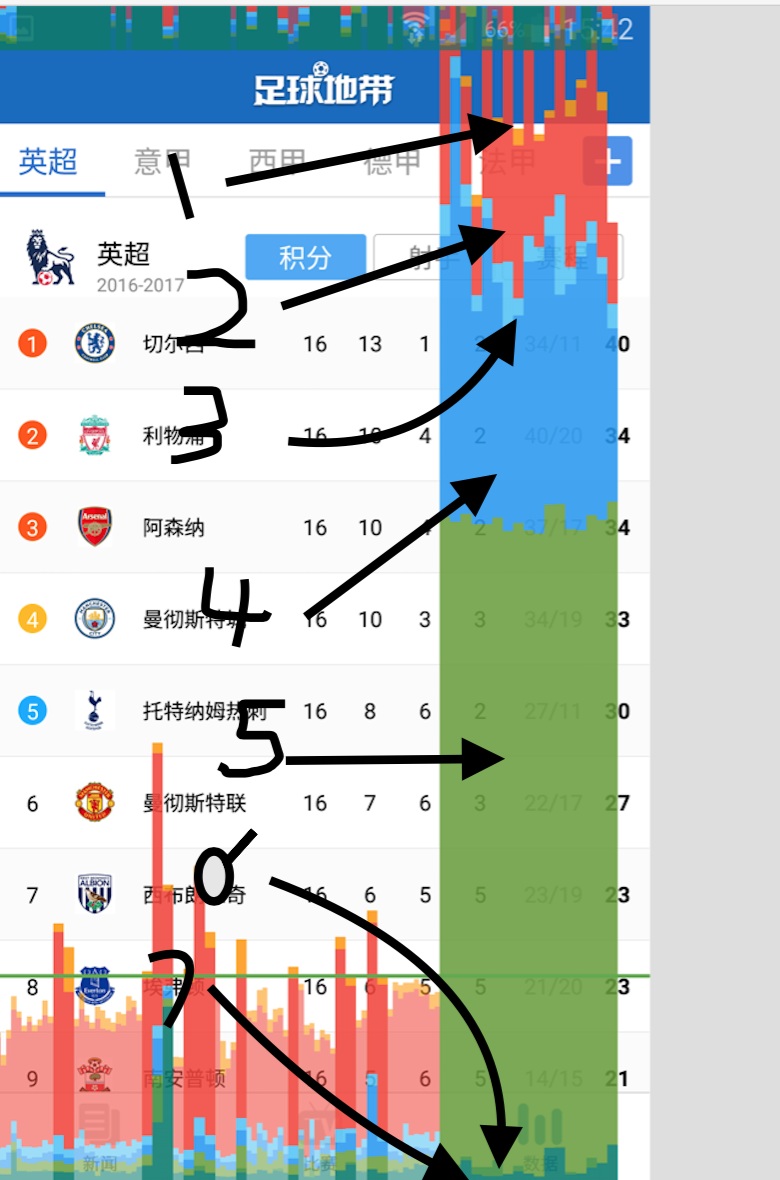
1. (这里由于色差原因，我分辨不出是Animation还是Input Handling)

Animation代表ObjectAnimation/ViewProperty/Transition等动画耗时；

Input Handling 表示GPU处理用户输入的耗时，或者说GPU在处理用户输入的回调耗时。

我在这里是重复点击 数据，基本不存在Animation的消耗，所以这里应该是处理Input Handling的耗时，点击的回调事件很简单，所以对应耗时也比较短；

1. MiscDelay~Misc Time：发生在两帧之间的耗时都可以用这个表示。



额，这里的话，还是讲得比较模糊，只是大概能理解这些概念，

有些概念理解起来都不好理解，所以先这样，这篇文档结ban合yun各种资料来理解，实用篇我再后边再写一篇。