# Day：GPU绘制流程及过度绘制

## 绘制原理

1、LayoutInflater将布局中的xml标签转化为对象，CPU经过measure、layout、draw将他们转化为多边形（Polygons）或纹理（Texture），GPU对多边形或纹理进行栅格化操作，栅格化后的数据写入帧缓冲区中等待显示器显示。

2、CPU和GPU通过图形驱动层连接。

3、GPU负责绘制帧，屏幕负责逐行扫描显示帧，两者速度不一定保持一致。

4、因此引入垂直同步机制，由系统每隔16ms发出一次VSync信号后，CPU进行计算，GPU进行绘制，以此来强制GPU的刷新率和屏幕刷新率同步。

5、理想状态下屏幕每秒展示60帧时人眼感受不到卡顿，动画会比较流畅，因此为保持应用流畅，我们需要在16ms的时间内完成绘制，尽量减少measure、layout、draw的时间。有以下解决方案：

* 精简布局层级
* 使用<include>标签重用布局，使用<merge>标签合并布局
* 使用ViewStub仅在需要时加载
* 删除无用的控件和布局
* 使用性能较好的ViewGroup，如Constraintlayout等
* onDraw方法中不要创建局部对象，会占用过多内存导致频繁gc
* onDraw方法中不要执行耗时操作及循环操作

## 过度绘制

1、过度绘制指：屏幕上的某个像素在同一帧的时间内被绘制了多次。会浪费CPU和GPU资源。

2、开发者工具可提高可视化过度绘制情况：真彩色-没有过度绘制；蓝色-过度绘制1次；绿色-过度绘制2次；粉色-过度绘制3次；红色-过度绘制4次或更多

3、过度绘制解决方案：

* 去掉Window的默认背景（在onCreate方法中或者在theme中）
* 去掉视图中不必要的背景
* 使用ViewStub仅在需要时加载
* 使用<include>标签重用布局，使布局更清晰明了；使用<merge>标签合并布局，减少层级
* 采用性能更好的布局来精简层级