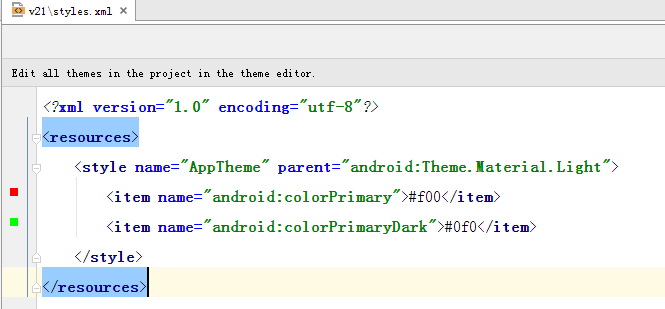
什么是Material Design

Material Design，中文名：质感设计，是由Google推出了全新的设计语言。说白了就是一套设计规范，给咱们提供了一些比较丰富的视觉效果

Material Design如何引用到项目中

新建项目的时候在res目录下创建values-21目录和styles.xml。在styles.xml引用Material Design主题



Material Design系统给咱们提供了3个默认的主题

1）android:Theme.Material

2）android:Theme.Material.Light

3）android:Theme.Material.Light.DarkActionBar

除了3种主题外还可以修改主题对应的颜色

1）状态栏颜色：colorPrimaryDark

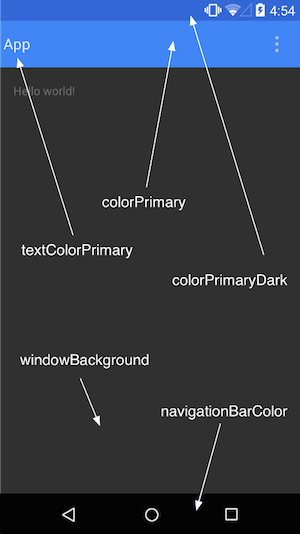
2）标题栏颜色：colorPrimary

3）主要文本颜色：textColorPrimary

4）次要文本颜色：textColorSecondary

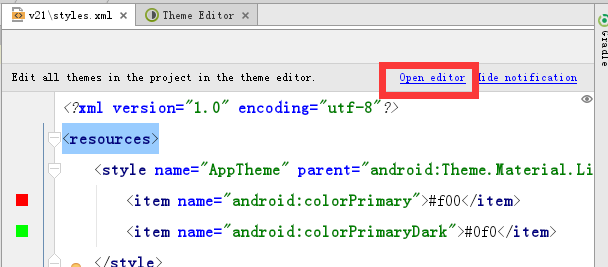
5）底部导航栏的颜色 ：navigationBarColor

6）背景色：windowBackground

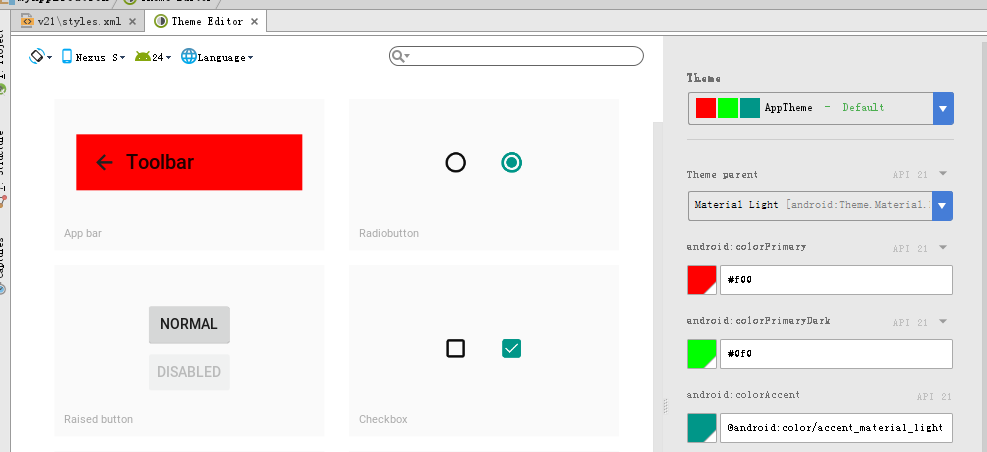


主题编辑器

第一步：



第二步：



代码切换主题

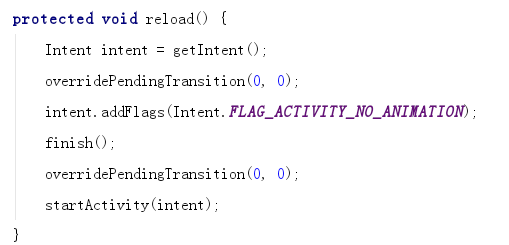
1）在vlaues-21目录下的styles.xml中定义主题

2）public static int current\_theme = -1;

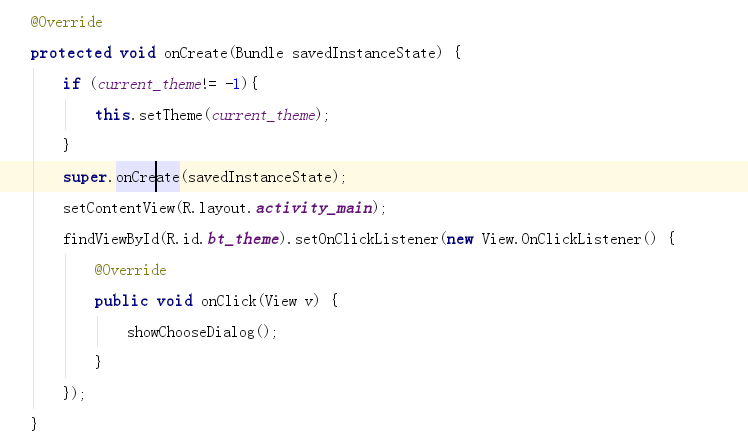
3）给current\_theme 设置成为要显示的主题样式

current\_theme = R.style.GreenTheme;

4）关闭页面重新打开页面



5）在activity重新初始化的时候将修改的主题设置成为新的主题

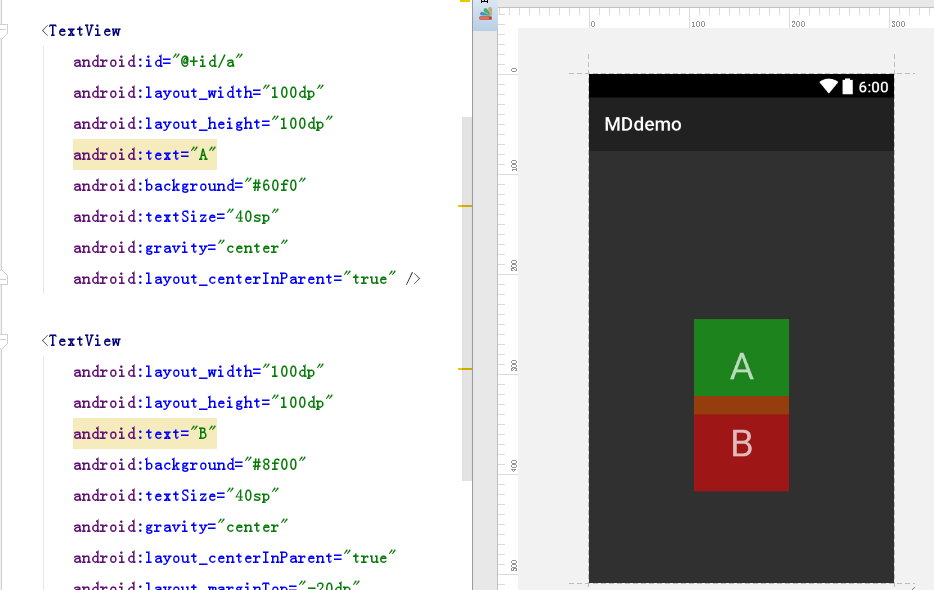


具体代码

阴影

在控件中声明属性**elevation**

在相对布局中会按顺序绘制布局，如下图所示



**但是如果加上elevation阴影属性，则是判断阴影的大小，如果A的阴影值大于B，则A会在上面。如果A的阴影数值大于B的则如下图：**



设置视图现实层级高度

1）elevation ：相对于父控件的高度

2）translationZ : 创建一个动画暂时的反应出View的高度值（elevation）变化

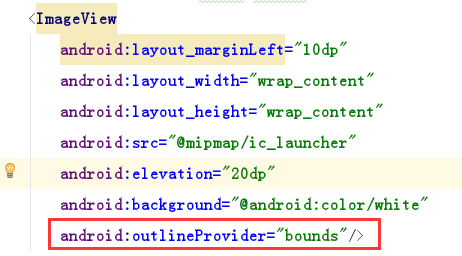
设置视图的阴影，outlineProvider属性

1）none

2）paddingBounds

3）bounds

4）background



使用ViewOutlineProvider裁剪阴影

注意：注意：并不所有的形状都可以剪裁，可通过OutlineProvider.canClip确认

1）裁剪圆角矩形

a） 构造ViewOutlineProvider，并实现它的getOutline（view , outline）

b） 调用outline的setRoundRect()方法，进行裁剪阴影

c） 将ViewOutlineProvider提供给视图的setOutlineProvider()

d） 将视图裁剪成对应的图形，需要设置视图的setClipToOutline为true

2）小正方形

a) 构造ViewOutlineProvider，并实现它的getOutline（view , outline）

b) 通过outline调用setRect()方法进行裁剪阴影

c) 将ViewOutlineProvider提供给视图的setOutlineProvider()

d) 将视图裁剪成对应的图形，需要设置视图的setClipToOutline为true

3）三角形

a) 构造ViewOutlineProvider，并实现它的getOutline（view , outline）

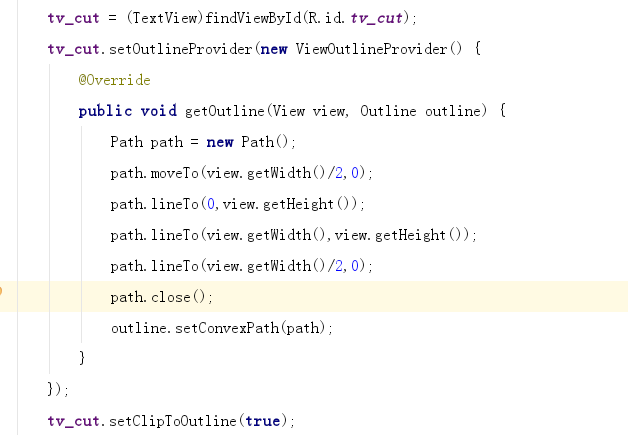
b) 通过Path绘制路径

c) 通过outline调用setConvexPath(path)进行裁剪阴影

d) 将ViewOutlineProvider提供给视图的setOutlineProvider()

e) 将视图裁剪成对应的图形，需要设置视图的setClipToOutline为true

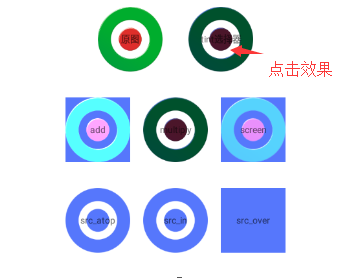
裁剪三角形失败，但是实际上阴影已经是三角形了





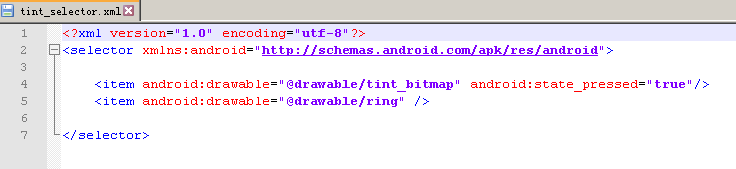
Tint混合图片

效果图：

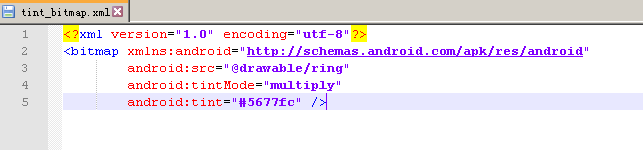


因为有点击效果所以本身是个selector

tint\_selector



tint\_bitmap



Tint混合图片制作方式

1）写一个xml，父标签是bitmap

2）通过src属性指定图片

3）通过tint设置染色色彩

4）通过tintMode指定染色模式(效果参照效果图)

a) add

b) src\_over

c) src\_atop

d) screen

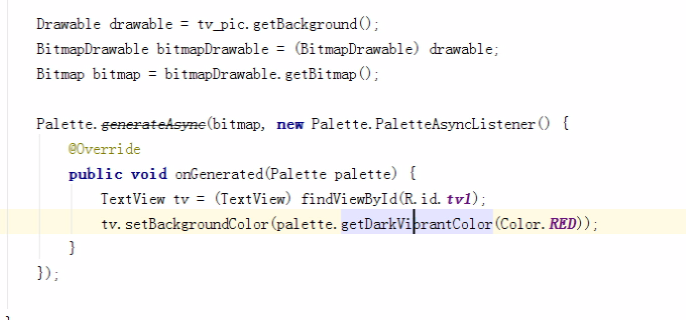
e) src\_in

f) multiply

取色器

注意在build.gradle中配置

compile 'com.android.support:palette-v7:23+'



在回调方法onGenerated(Palette palette) 中可以通过palette对象获取6种不同颜色

1）palette.getVibrantColor(Color.BLACK)//Vibrant 充满生机

2）palette.getDarkVibrantColor(Color.BLACK)//Vibrant dark 暗的充满生机

3）palette.getLightVibrantColor(Color.BLACK)//Vibrant light 亮的充满生机

4）palette.getMutedColor(Color.BLACK)//Muted 柔和

5）palette.getDarkMutedColor(Color.BLACK)//Muted dark 暗的柔和

6）palette.getLightMutedColor(Color.BLACK)//Muted light 亮的柔和

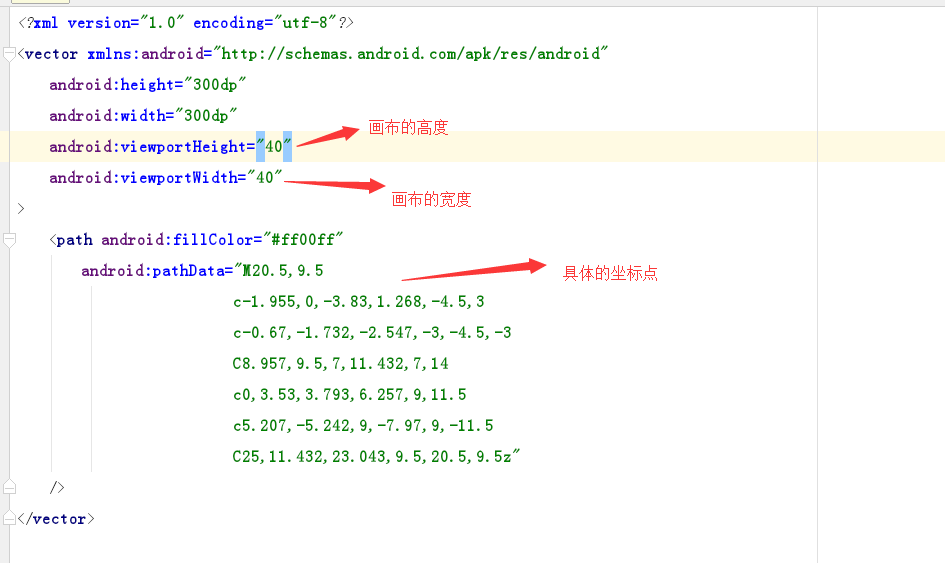
矢量图

矢量图的特点：

1）文件小,图像中保存的是线条和图块的信息,所以矢量图与分辨率和图像大小无关,只与图像的复杂程度有关,图像文件所占的存储空间较小

2）图像可以无级缩放,对图形进行缩放,旋转或变形操作时,图形不会产生锯齿效果

利用xml画矢量图：



其中数据的获取可以在<http://editor.method.ac/>这个网站上画出图形然后查看数据

波纹动画

使用：

1）android:background = “?android:selectableItemBackground”

2）android:background = “?android:selectableItemBackgroundBorderless”

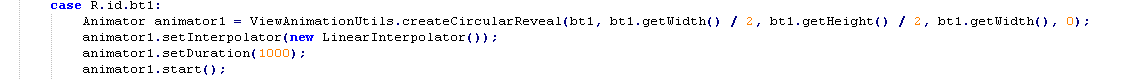
低版本兼容：

<https://github.com/traex/RippleEffect>

自定义波动动画：

1）通过ViewAnimationUtils给我们提供了createCircularReveal(目标view , 中心点x , 中心点Y，起始位置，结束位置)方法来实现揭示动画

2） 返回一个动画对象，然后给动画设置属性开启动画



动画选择器

