# 混合富文本

// 通过组合富文本方式，来实现一组字符串，多种颜色color，多种字体ttf  
SpannableStringBuilder builder = new SpannableStringBuilder();  
// 动态添加富文本  
SpannableString textSpanned1 = new SpannableString("Hello World");  
textSpanned1.setSpan(new TypefaceSpan("富文本组合"),  
 0, 3, Spanned.*SPAN\_EXCLUSIVE\_EXCLUSIVE*);  
  
SpannableString textSpanned2 = new SpannableString("Hello World");  
textSpanned1.setSpan(new TypefaceSpan("富文本组合"),  
 0, 3, Spanned.*SPAN\_EXCLUSIVE\_EXCLUSIVE*);  
// 添加组合  
builder.append(textSpanned1).append(textSpanned2);  
// 设置富文本  
new TextView(this).setText(builder);

# 关于编码集和渠道的理解

## 渠道的理解

android **{** compileSdkVersion 30  
 buildToolsVersion "30.0.2"  
 defaultConfig **{  
 //**。。。  
 flavorDimensions "default"  
 **}** productFlavors **{**

**// hk渠道** hk **{  
 }  
// macao渠道** macao **{  
 }  
 }**

buildTypes **{**

**// 构建类型，默认是debug和release** release **{** minifyEnabled false  
 proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'  
 **}  
}**

**//。。。**

**}**

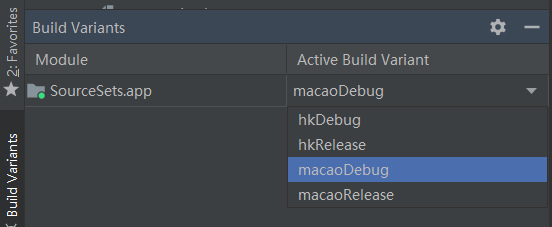
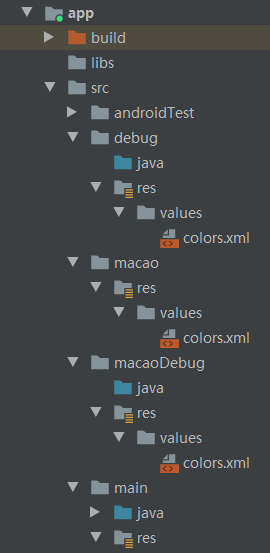
变体：变种+类型，如图：hkDebug，hkRelease，macaoDebug，macaoRelease

构建类型：默认存在debug和release，其他需要指明，在buildTypes里面配置

变种：渠道，如图hk和macao

主源码集：main

## 代码合并规则



加载的优先级：变体中的列表 > 类型 > 变种 > main

清单文件manifest，values和res和asset，允许同名的资源，可以替换！

java中的类如果重复，会报Error错误！类不能重复

（如上图，在scr目录下，定义了多个源码集colors）

<?xml version="1.0" encoding="utf-8"?>  
<resources>  
 <color name="root\_bg">#FFff00FF</color>  
</resources>

当切换到macaoDebug的时候，由于macaoDebug是变体，为最高等级，则优先加载macaoDebug中的颜色

当切换到hkDebug的时候，由于没有hkDebug，但是存在debug，则优先加载debug中的颜色

## 源码集的理解

默认的源码集：app\src\变种\相关目录

告诉gradle应该从哪个位置来合并代码，生成apk！

默认位置：如下

macao

-----

Compile configuration: macaoCompile

build.gradle name: android.sourceSets.macao

Java sources: [app\src\macao\java]

Manifest file: app\src\macao\AndroidManifest.xml

Android resources: [app\src\macao\res]

Assets: [app\src\macao\assets]

AIDL sources: [app\src\macao\aidl]

RenderScript sources: [app\src\macao\rs]

JNI sources: [app\src\macao\jni]

JNI libraries: [app\src\macao\jniLibs]

Java-style resources: [app\src\macao\resources]

sourceSets **{** hk **{  
  
 }** macao **{** java.srcDirs = ['aa'] // = ['aa'] 这个是完全替换  
 res.srcDirs('bb') // ('bb') 这个是在源目录上，追加目录  
 **}  
}**

macao

-----

Compile configuration: macaoCompile

build.gradle name: android.sourceSets.macao

Java sources: [app\aa] // 完全替换

Manifest file: app\src\macao\AndroidManifest.xml

Android resources: [app\src\macao\res, app\bb] // 追加

Assets: [app\src\macao\assets]

AIDL sources: [app\src\macao\aidl]

RenderScript sources: [app\src\macao\rs]

JNI sources: [app\src\macao\jni]

JNI libraries: [app\src\macao\jniLibs]

Java-style resources: [app\src\macao\resources]

大多数情况下，不需要改变和调整源码集位置

可能用得到的是把jniLibs的默认引用位置，改到libs

## Gradle其他说明

name：build type的名字

applicationIdSuffix：应用id后缀

versionNameSuffix：版本名称后缀

debuggable：是否生成一个debug的apk

minifyEnabled：是否混淆

proguardFiles：混淆文件

signingConfig：签名配置

manifestPlaceholders：清单占位符

shrinkResources：是否去除未利用的资源，默认false，表示不去除。

zipAlignEnable：是否使用zipalign工具压缩。

multiDexEnabled：是否拆成多个Dex

multiDexKeepFile：指定文本文件编译进主Dex文件中

multiDexKeepProguard：指定混淆文件编译进主Dex文件中