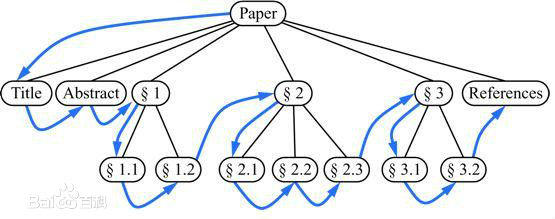
2017/1/15 树的概况与测试代码

今天我要尝试在Android Studio上开始写一个数据结构——多叉树。



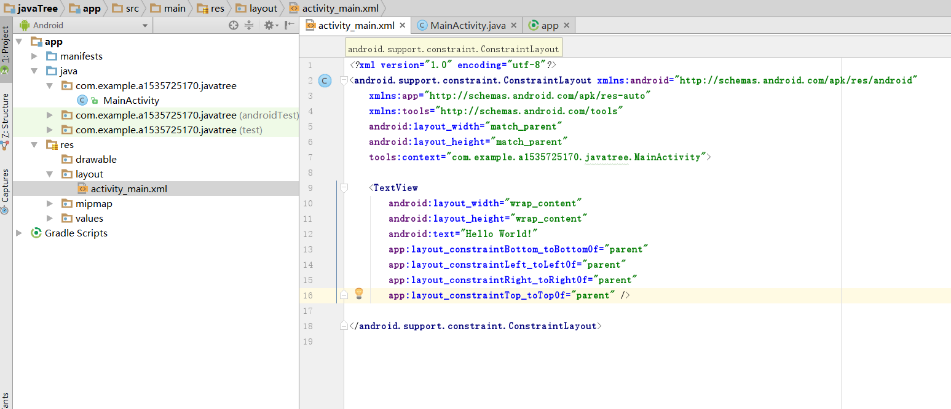
先简单介绍一下怎么用多叉树，多叉树有根节点，内部节点和叶子节点，我打算用根节点记录这棵树的概况，内部节点用于放置逻辑语句，叶子节点用于放置命令。用多叉树的好处在于可以有效地管理命令执行的顺序和有效地储存图形化编程的结果。

这次我新开了一个项目javaTree，专门用来测试内部数据结构的逻辑，目标还是比较简单，手动在代码里伪造一颗多叉树，界面上只有一个按钮Button和显示处TextView，通过点击按钮触发TextView的显示，来看看程序内部的运转状况。

刚生成的文件如果不能正常运行，别忘了按照工作日志中的第一个问题解决方案，把工程最低sdk改到18（可能不一定是18，一般是18）

在构思时，我觉得在这一章里面比较大的难题在于不知道在哪里写代码（当然最大的难题还是树的逻辑实现）。你可以先不看我下面的内容，好好想一想。

先进入layout->activity\_main.xml，这个是MainActivity对应的布局文件，和上一章中factivity和FActivity的关系是一样的。以下是初始状态:



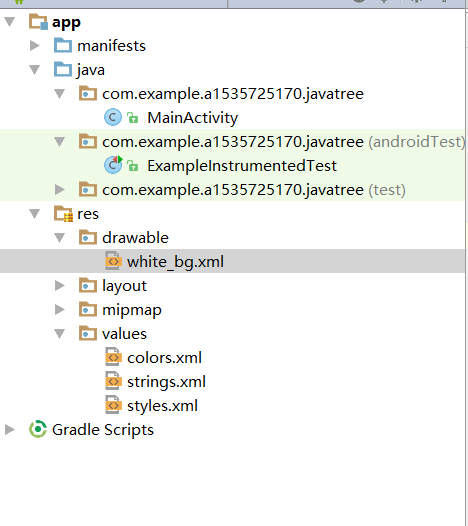
这里已经有一个textView了，显示的是“helloworld”，我们要做的是让它显示变量，可以通过在MainActivity里不断的setText来实现。

在这里要做的是给它增加一个id，这样才好在MainActivity里面引用它。



我想要有个稍微好看一点的textView

在drawable文件夹里新建一个XML文件（New->XML->Values XML files）这个名字取得不是很走心，因为这只是用来测试的项目，正式写的时候还是要认真取名字。



按照如下代码，分别设置圆角，渐变色和框

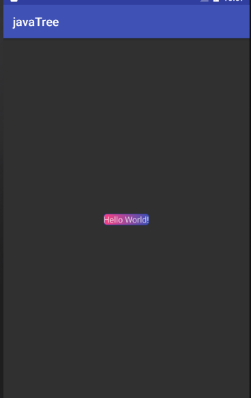
这里的颜色可以在values->colors里面自己定义的



然后再设置textView属性的地方，增加background属性来引入这个样式。



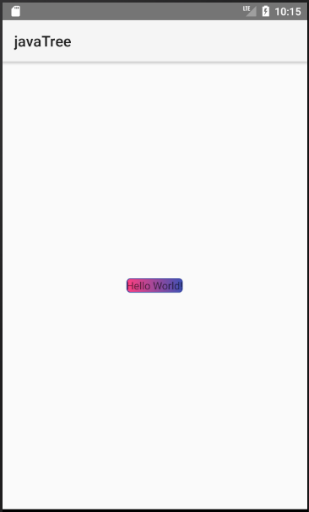
这样就可以做出一个稍微好看一点的textView了



感觉黑色的背景太丑，我们可以在manifest（AndroidManifest.XML）的activity里设置theme，就是中间很明显给你看的

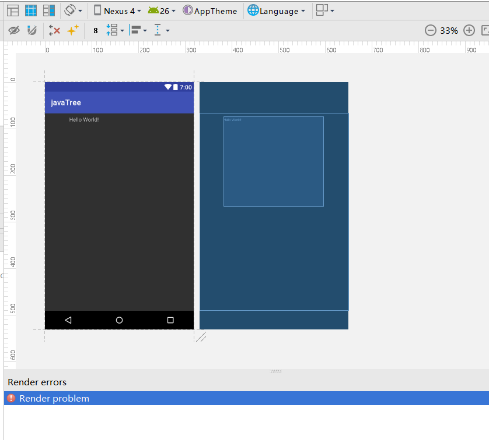


我们得到了效果：



然后有一个小小的问题

Android studio中Preview出现render problem问题（我把textView变大并且添加了约束，不用管）



解决方案如下<http://blog.csdn.net/mygfz/article/details/53058165>或者可以不管的（因为只是不能预览）

这只是今天的开胃小菜，其实之后也不一定用的到，只是多多少少想要秀一波操作。

然后初始化一个按钮（在design模式下直接拖拽就行了），改变id和text

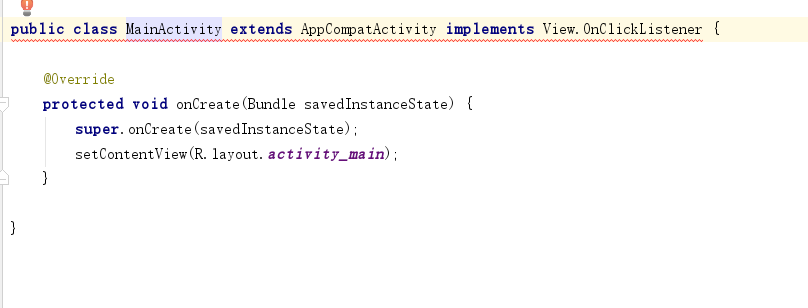


下面我们实现按钮的点击事件OnClickListener

在MainActivity.java里

需要改动这一条C:\Users\1535725170\AppData\Roaming\Tencent\Users\1535725170\QQ\WinTemp\RichOle\XYG%8TO)R]Y)OL3K(61H[68.png

变成这样（这里面View.是自动添加的），但是会飘红，因为需要在里面重写OnClick函数



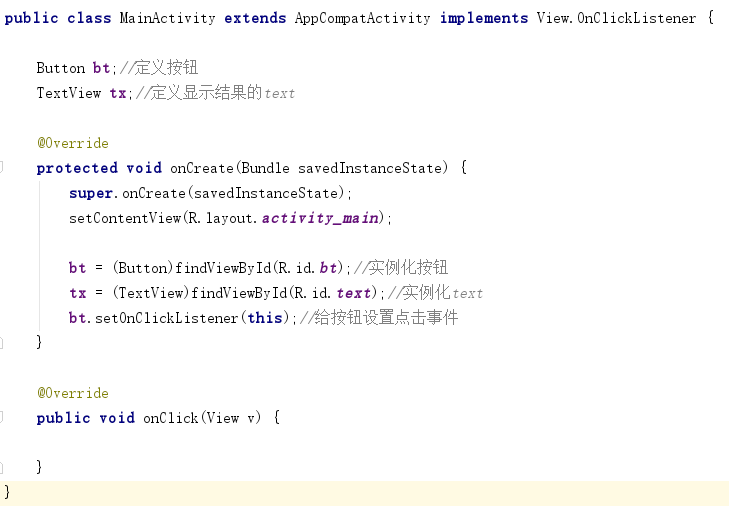
点击左边的红灯泡或者报错提示，红灯泡里有implement methods，点它，也可以手打函数onClick，这个函数是看点击哪个按钮时做什么事的，很重要



首先我们要定义按钮和text（常规操作）

然后实例化这两个控件

给按钮设置点击事件



接下来我们就可以在onClick里面实现“业务逻辑”了

一般的用法是在这个函数里

switch（v.getId()）{

case bt:

做什么什么事

Break;

case xx:

做什么什么事

Break;

。。。。。。

Default：

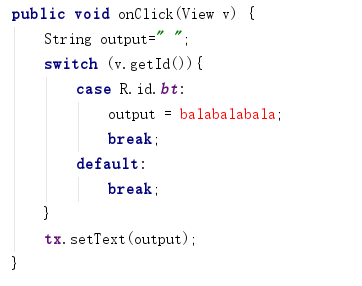
Break;

}

先在里面声明局部变量String output,注意这里String的S是大写

虽然这个测试程序只有一个按钮，但是为了规范起见，也套用这个格式

最后用setText来输出



如果有需要用到之前的输出的，可以这样做：

把String output=“ “改成String output=**tx**.getText().toString();

这样我们的显示框架就搭好了，以后要测试什么函数也可以在这里面测试了。