**中山大学移动信息工程学院本科生实验报告**

**（2017学年秋季学期）**

|  |  |  |  |
| --- | --- | --- | --- |
| 年级 | **15** | 专业（方向） | **移动信息工程** |
| 学号 | **15352320** | 姓名 | **王诗语** |
| 电话 | **15626007477** | Email | **1532079717@qq.com** |
| 开始日期 | **2017.09.23** | 完成日期 | **2016.09.23** |

课程名称： 移动应用开发 任课教师：郑贵锋

【实验题目】

——基本 UI 界面设计

【实验工具】

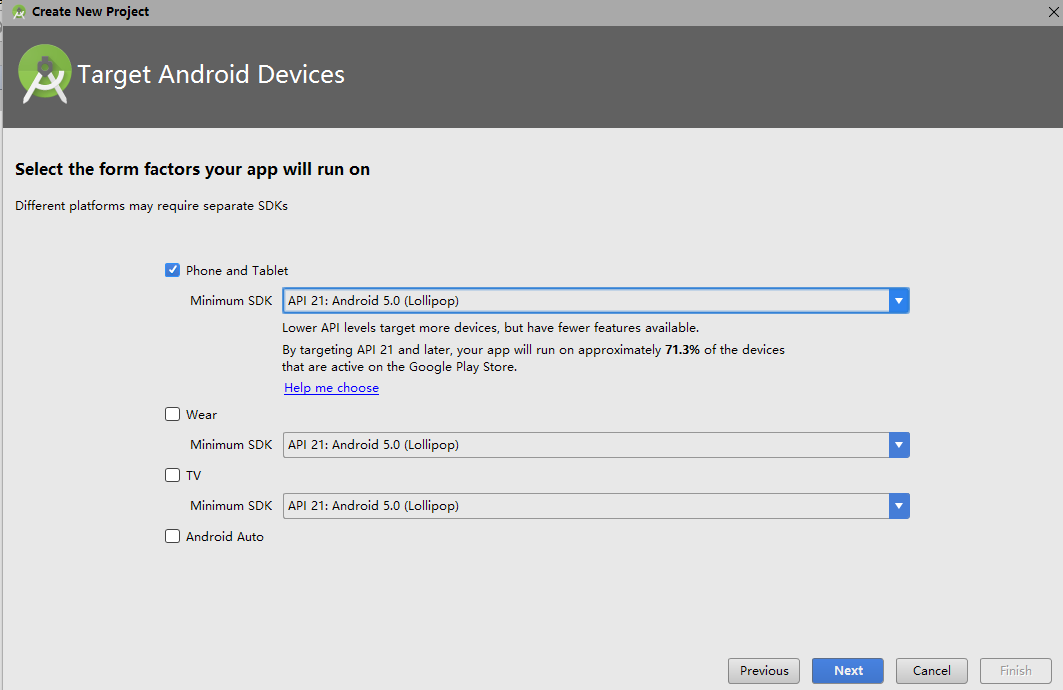
Android studio 2.3.3：constraint layout

【实验目的】  
1.熟悉 Android Studio 开发工具操作   
2.熟悉 Android 基本 UI 开发,并进行 UI 基本设计

【实验过程】

1. **环境搭建**

新建android project时，SDK的选择上一般选择的最小SDK是API15:Android 4.0.3，这一项可以根据自己的手机配置来选择，具体信息可以在手机about中查看，顺便，手机的分辨率也可以一起记录一下。



1. **Constraint Layout简介**

约束布局由Google I/O在2016年推出，虽然发布时间较短，但从支持力度而言, 将成为主流布局样式,。而且因为约束布局减少了布局的层级, 很大程度上优化了渲染性能，所以将完全代替其他布局。

Constraint Layout根据布局中的其他元素或视图, 确定View在屏  
幕中的位置, 受到三类约束, 即**其他视图, 父容器(parent), 基准线  
(Guideline)**.设置约束的基本语法如下：  
layout constraint[本源位置]\_[目标位置]="[目标ID]“  
例如语句

app:layout\_constraintBottom\_toBottomOf=“parent”（即父容器）

的效果就是把当前View的底部对齐到Constraint Layout的底部.

1. **UI实现**

UI界面效果如下图。



在constraintLayout的design界面中每个部件都可以通过拖动以及连接约束来定位。如果是代码，每个控件都有自己的ID，当你要将两个控件彼此间建立约束的话，只要改变上述例子中的parent为控件ID就能实现。

代码中每设置一次约束，如连接到图片底部：

**app:layout\_constraintTop\_toBottomOf="@+id/imageView"**

距离的设置（如果需要）就要配套写一行：

**android:layout\_marginTop="20dp"**

（margin这个概念我留在最后的心得里与padding一起比较）

左方Palette窗口中，可以选择需要的东西拖入容器中，相当于建立好了一个控件。

点击一个控件，在右端Properties窗口中，可以更改长宽、文字等属性，上方正方形试图可以通过设置与约束部件的距离来调整当前控件在view中的位置。一般默认View的长宽采用wrap\_content，如果采用match\_content，则会自动匹配到与父容器等长（宽）。在Properties窗口最下方找到View all properties，可以对控件各种详细的属性，如内边距，颜色，背景框等进行设置。

但是很多人反映这个design界面常常碰一下就歪，异常地玄学，所以大都采取了直接打代码的方式，虽然慢了点不过确实很稳健。最好的方法当然是两者的结合，不过这大概需要经验的积累，才能清楚什么操作在哪个界面比较方便。

至少，上面我说的那些在design界面的拖拽和属性设置功能可以帮助自动生成大量繁琐的代码。

下表是本次基本UI实现所需要的控件。

|  |  |
| --- | --- |
| 效果 | 控件 |
| 标题 | TextView |
| 文本输入框 | EditText |
| 图片 | ImageView |
| 单选框 | RadioGroup |
| 按钮 | Button |

图片文件我这次实验里我放在mipmap下，当然这与放在drawable下是有不同的，分辨率就是一个问题，不过中大图标本身就那么点大。如果只是小图标的话建议放在mipmap下，mipmap新建图片时会自动生成适合图标使用的大小并提供五种分辨率，drawable则是直接导入图片，使用情况有所不同。

EditText中学号和密码分别需要选为number和textPassword类型，效果是只能输入数字和密码显示为实心圆点。关于输入框，可以使用

**android:ems="10"**

设置TextView或者EditText的宽度为10个字符的宽度。当设置该属性后，控件显示的长度就为10个字符的长度，超出的部分将不显示。

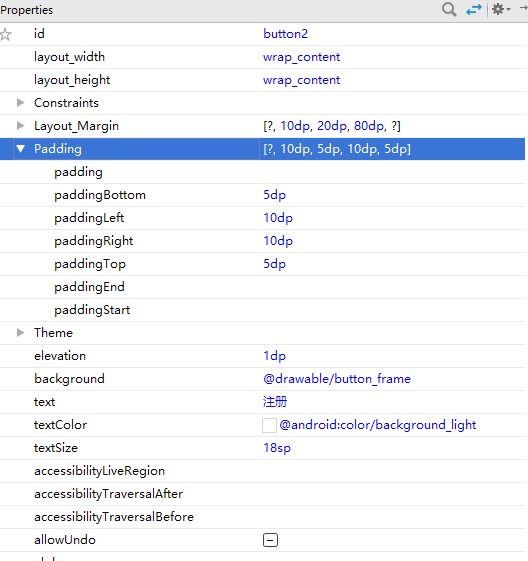
单选按钮需要建立一个RadioGroup,一个Group中可以包含若干个RadioButton，可以使用语句：

**android:orientation="horizontal"**

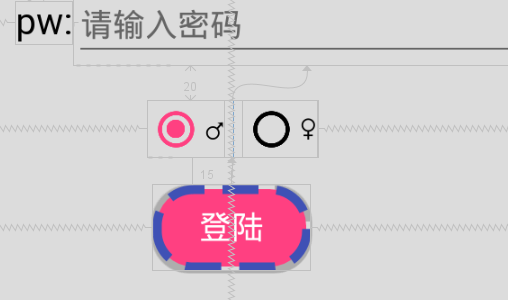
来决定各个RadioButton是水平排列还是垂直排列。

整个Group的效果在容器中体现为只能选择一个选项。很方便的一点是要改变这一套选择按钮在布局中的位置只需要调整Group的位置即可。

Button的属性设置建议在Properties窗口下进行，包括padding的设置，背景的导入，都十分方便。这里矩形框可能看不出来padding的作用，实际上padding可以定位按钮中文本的位置，比如做了一个奇形怪状的button，就需要为文本挑一个顺眼的位置放着。



关于自定义Button的背景边框，当需要将一个 button 设置为圆角矩形时，光设置 button 的属性是达不到效果的，需要定义一个背景边框来达到这个效果。



在 drawable 文件夹下新建一个 Drawable resource file，填写 file name，然后把自动生成的 selector 标签改为 shape，具体怎么使用参见这篇教程：

<http://blog.csdn.net/sysukehan/article/details/52022307>

【实验结果】

安卓手机展示：



【心得体会】

1. **关于Start/End和Left/Right**

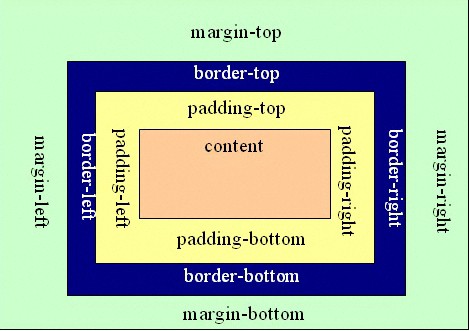
  一般情况下，View开始部分就是左边，但是有的语言目前为止还是按照从右往左的顺序来书写的，例如阿拉伯语，在Android  4.2系统之后，Google在Android中引入了RTL布局，更好了支持了由右到左文字布局的显示，为了更好的兼容RTL布局，google推荐使用MarginStart和MarginEnd来替代MarginLeft和MarginRight，这样应用可以在正常的屏幕和由右到左显示文字的屏幕上都保持一致的用户体验。

参考http://blog.csdn.net/zhufuing/article/details/40181815

1. **关于padding和magin**

概念：padding是站在父view的角度描述问题，它规定它里面的内容必须与这个父view边界的距离。margin则是站在自己的角度描述问题，规定自己和其他（上下左右）的view之间的距离。

一张图可以看懂：



1. **控件居中**
   1. 如果是水平线上就一个View,则将View与parent两边建立左右约束就会自动居中，不放心的话可以右键单击View，选择Central Horizontally（in parent），体现在代码里就是

**app:layout\_constraintHorizontal\_bias="0.5"**

0.5指的是居中，是一个比例值，试着改成0.1或0.8，View就会偏左或偏右。如下：一个0.3的偏移



②利用guideline

<**android.support.constraint.Guideline  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"   
 android:id="@+id/myguide"  
 android:orientation="vertical"  
 app:layout\_constraintGuide\_begin="22dp"** />

可以在距layout左边界22dp处画一条竖直的guideline

③多个View并列，可以建立链式连接：

通过 app:layout\_constraintHorizontal/Vertical\_chainStyle 设置链接的样式。

其中，一种可自动居中的链接为packedchain，将chainStyle设置为“packed”，关于packedchain的定义如下：

CHAIN\_PACKED -- the elements of the chain will be packed together. The horizontal or vertical bias attribute of the child will then affect the positioning of the packed elements

可见设置bias可以实现居中。

**4．小结**

Constraint Layout这种布局方式比起相对布局和线性布局，灵活度更高。这次的UI完成的过程也比较顺利，感觉安卓应用开发没有想象中那么枯燥，在看到手机上展示出自己界面的时候也挺有成就感的。所以做完之后也着手查资料打算学习一下按钮点击的实现等，毕竟挺想做出一个自己的app的。