中山大学移动信息工程学院本科生实验报告

（2017 年秋季学期）

课程名称：移动应用开发　　任课教师： 郑贵锋

年级	2015	专业（方向）	移动互联网
学号	15352286	姓名	任萌
电话	13763361232	Email	352600801@qq.com
开始日期	2017 年 9 月 22 日	完成日期	2017 年 9 月 25 日

一、 实验题目

基本 UI 界面设计

二、 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

三、 实现内容

只用一个 ConstraintLayout 布局实现一个 Android 应用的启动界面，界面效果图：

使用到的布局和控件：ConstraintLayout、TextView、EditText、Button、ImageView、RadioGroup、RadioButton。

使用到的代表性函数：

函数名称	功能
android:layout_width	设置组件的宽度
android:layout_height	设置组件的高度
android:text	设置文本内容
android:textColor	设置文本颜色
android:id	设置控件 id 号
android:textSize	设置文本大小
app:layout_constraintLeft_toLeftOf	设置控件的左边与...的左边对齐
android:layout_marginTop	设置控件距顶部控件的距离
app:srcCompat	导入图片
android:hint	设置输入框的提示内容
android:inputType	设置输入框输入格式
android:orientation	设置组件排列方向
android:checked	是否被设置为选中
app:layout_constraintHorizontal_chainStyle	设置链条内组件分布类型

四、 课堂实验结果

（1） 实验截图

结果分析：整体布局效果基本符合实验文档的要求，同时在学号和密码的输入框中都分别实现了只允许输入数字和密码显示为*。

（2） 实验步骤以及关键代码

Step 1 创建一个 Android Studio 的 project，创建布局文件 activity_ex1.xml

Step 2 为了方便后期对布局代码进行 debug 和修改，故预先定义了 colors、dimen、string 等文件，对规定文本内容和属性等进行了定义，在布局文件中利用@进行引用

Step 2.1 定义颜色文件 colors.xml

colorText：文本颜色，为黑色

colorButton：按钮文本颜色，为白色

colorPrimary：按钮背景颜色，为深蓝#3F51B5

```
<resources>
    <color name="colorPrimary">#3F51B5</color>
    <color name="colorPrimaryDark">#303F9F</color>
    <color name="colorAccent">#FF4081</color>
    <color name="colorText">#000000</color>
    <color name="colorButton">#ffffff</color>
</resources>
```

Step 2.2 定义尺寸 dimen.xml

normalText：普通文本大小，20sp

normalDistance：普通控件间间距，20dp

littleDistance：较小间距，10dp

smallText：小文本大小，如按钮和提示框文本，18sp

```
<resources>
    <dimen name="normalText">20sp</dimen>
    <dimen name="normalDistance">20dp</dimen>
    <dimen name="littleDistance">10dp</dimen>
    <dimen name="smallText">18sp</dimen>
</resources>
```

Step 2.3 定义字符串内容 string.xml

```
<resources>
    <string name="app_name">Ex1</string>
    <string name="title">中山大学学生信息系统</string>
    <string name="user">学号：</string>
    <string name="key">密码：</string>
    <string name="input_user">请输入学号</string>
    <string name="input_key">请输入密码</string>
    <string name="student">学生</string>
    <string name="teacher">教职工</string>
    <string name="login">登录</string>
    <string name="register">注册</string>
    <string name="college">学院：</string>
</resources>
```

Step 3 利用 TextView 实现标题

设置组件的宽度和高度与内容适配，设置标题内容为“中山大学学生信息系统”，标题字体大小为 20sp，设置标题 id 为 title，利用 layout_marginTop 实现与顶部距离 20dp，利用 layout_constraintLeft_toLeftOf=”parent”和 layout_constraintRight_toRightOf=”parent”将部件的左端与右端分别与父组件对齐，实现居中显示。

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/title"
    android:textColor="@color/colorText"
    android:textSize="@dimen/normalText"
    android:id="@+id/title"
    android:layout_marginTop="@dimen/normalDistance"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent" />
```

Step 4 利用 ImageView 实现中大 logo 显示

首先将图片导入到 mipmap 中，并命名为 sysu，利用 srcCompat 实现对图片的调用。然后设置组件的宽度和高度与内容适配，设置图片 id 为 sysu，设置图片与标题的间距为 20dp，居中，让图片的顶端和标题底部对齐。

```
<ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/sysu"
    app:srcCompat="@mipmap/sysu"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintLeft_toLeftOf="parent"
    android:layout_marginTop="@dimen/normalDistance"
    app:layout_constraintTop_toBottomOf="@+id/title" />
```

Step 5 利用 TextView 和 EditText 实现输入框

输入提示部分：首先设置组件的宽度和高度与内容适配，设置提示内容为“学号：”（“密码：”），颜色为黑色，id 为 user（key），文字大小为 18sp，距离左端 20dp，第一行距离上端 logo30dp，第二行距离第一行 20dp。

输入部分：输入框整体距屏幕右边间距 20dp，上下两栏间距 20dp，手动设置输入框宽度为 250dp，高度与内容适配，设置输入提示文字“请输入学号”（“请输入密码”），限定输入类型 number（textPassword），设置 id 为 input_user（input_key）。

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/user"
    android:textColor="@color/colorText"
    android:id="@+id/user"
    android:textSize="@dimen/smallText"
    app:layout_constraintLeft_toLeftOf="parent"
    android:layout_marginLeft="@dimen/normalDistance"
    android:layout_marginTop="30dp"
    app:layout_constraintTop_toBottomOf="@+id/sysu" />
```

```
<EditText
    android:layout_width="250dp"
    android:layout_height="wrap_content"
    android:id="@+id/user_edit"
    android:hint="@string/input_user"
    android:maxLength="8"
    android:inputType="number"
    android:textSize="@dimen/smallText"
    app:layout_constraintLeft_toRightOf="@+id/user"
    app:layout_constraintBottom_toBottomOf="@+id/user"
    app:layout_constraintTop_toBottomOf="@+id/sysu"
    android:layout_marginTop="@dimen/normalDistance"
    android:layout_marginRight="@dimen/normalDistance"
    app:layout_constraintVertical_bias="0.545"
    app:layout_constraintRight_toRightOf="parent"
    android:layout_marginEnd="@dimen/normalDistance" />
```

Step 6 利用 RadioGroup 和 RadioButton 实现单选按钮

利用 RadioGroup 设置整体居中，与顶部控件输入框相距 20dp，且内部组件按水平方向排列。设置两个单选按钮宽度和高度与内容适配，字体大小 18sp，间距 10dp，利用 checked 属性实现默认选中效果。

```
<RadioGroup
    android:id="@+id/buttonGroup"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    app:layout_constraintTop_toBottomOf="@+id/xueyuan"
    android:layout_marginTop="@dimen/normalDistance"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintLeft_toLeftOf="parent">

    <RadioButton
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/student"
        android:text="@string/student"
        android:textSize="@dimen/smallText"
        android:checked="true" />
```

Step 7 利用 Guideline 和 button 实现底部按钮分布

Step 7.1 定义自定义背景边框

在 drawable 文件夹下新建一个 Drawable resource file，填写文件名后把自动生成的 selector 标签改为 shape。利用 radius 设置圆角半径为 10dp，利用引用将背景色设置为#3F51B5，在 padding 中设置按钮背景框左右边框与文字间距 10dp，上下边框与文字间距 5dp。

```xml
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <corners
        android:radius="10dp"/>
    <solid
        android:color="@color/colorPrimary"/>
    <padding
        android:top="5dp"
        android:bottom="5dp"
        android:left="10dp"
        android:right="10dp"/>
</shape>
```

Step 7.2 统一定义按钮的形式

新建一个 styles.xml 文件，统一将两个按钮的宽度和高度设置为与内容适配，文字大小为 18sp，与上方控件间距 20dp，背景为 shape，并分别设置文本内容。

```xml
<style name="button1_style">
    <item name="android:layout_width">wrap_content</item>
    <item name="android:layout_height">wrap_content</item>
    <item name="android:text">@string/login</item>
    <item name="android:textSize">@dimen/smallText</item>
    <item name="android:textColor">@color/colorButton</item>
    <item name="android:background">@drawable/shape</item>
</style>

<style name="button2_style">
    <item name="android:layout_width">wrap_content</item>
    <item name="android:layout_height">wrap_content</item>
    <item name="android:text">@string/register</item>
    <item name="android:textSize">@dimen/smallText</item>
    <item name="android:textColor">@color/colorButton</item>
    <item name="android:background">@drawable/shape</item>
</style>
```

Step 7.3 利用 Guideline 实现整体居中

首先在可视化界面内增加位于中间位置的垂直引导线。

```xml
<android.support.constraint.Guideline
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/guideline"
    app:layout_constraintGuide_begin="192dp"
    android:orientation="vertical" />
```

然后利用 app:layout_constraintRight_toLeftOf 和 app:layout_constraintLeft_toRightOf 函数将两个按钮组合成一个链条，并且设置它们与引导线相距 5dp，从而实现整体居中且按钮间隔 10dp 的效果。

```xml
<Button
    android:id="@+id/button1"
    style="@style/button1_style"
    app:layout_constraintTop_toBottomOf="@+id/buttonGroup"
    android:layout_marginTop="@dimen/normalDistance"
    app:layout_constraintRight_toLeftOf="@id/button2"
    app:layout_constraintRight_toRightOf="@id/guideline"
    android:layout_marginRight="5dp"
    app:layout_constraintHorizontal_chainStyle="spread" />

<Button
    android:id="@+id/button2"
    style="@style/button2_style"
    app:layout_constraintBottom_toBottomOf="@+id/button1"
    app:layout_constraintLeft_toRightOf="@id/button1"
    app:layout_constraintLeft_toLeftOf="@id/guideline"
    android:layout_marginLeft="5dp"/>
```

（3） 实验遇到困难以及解决思路

Q1：不知道如何将底部的两个按钮打包在一起。

解决方法：一开始为了实验效果，直接设置两个按钮上端和下端分别对齐，设置 button2 距离 button1 为 10dp，然后手动将两个按钮移动到了屏幕中心。虽然从最终效果来看差不多，但是还是没有用到 chain 的方法。后来，我再次仔细研读了实验文档，发现可以利用 app:layout_constraintRight_toLeftOf="@id/button2"和 app:layout_constraintLeft_toRightOf="@id/button1"的双向链接实现两个控件的绑定。

Q2：在将两个按钮组合成一个链条后，不知道该怎样调整它们之间的间距。

解决方法：在按钮成功打包后，我又被调整间距难住了。不论我怎么设置按钮的 marginLeft 和 marginRight，布局都没有发生任何变化。后来我发现在 ConstraintLayout 中，可以借助引导线来实现对齐，于是我在中央位置增加了一条垂直引导线，并让两个按钮处在引导线两侧且距离引导线 5dp，这样便实现了居中且相距 20dp 的效果。

五、 课后实验结果

（1） 实验内容

① 增加了对输入框内输入内容的长度限定

② 增加了对学院的下拉选择

（2） 实验截图

结果分析：在原来的功能基础上增加了对学院的选择，点击选择框会出现设定好的五个学院，选定某个学院后显示变为被选定学院名称。

（3） 实验步骤以及关键代码

Step 1 增加对输入框内输入内容的长度限定

使用 maxLength 即可。

```
android:maxLength="16"
```

Step 2 用下拉框实现对学院的选择

Step 2.1 新建 arrays.xml 文件，用来设计下拉框中的选择项。

```
<resources>
    <string-array name="college">
        <item>数据科学与计算机学院</item>
        <item>传播与设计学院</item>
        <item>管理学院</item>
        <item>岭南学院</item>
        <item>中山大学医学院</item>
    </string-array>
</resources>
```

Step 2.2 利用 Spinner 布局添加下拉框控件，和学号、密码输入框部分设计类似，使其距离右边框 20dp，距离上面的密码输入框 20dp，和父控件右对齐。

```
<Spinner
    android:id = "@+id/college"
    android:layout_width = "wrap_content"
    android:layout_height ="wrap_content"
    android:entries="@array/college"
    app:layout_constraintLeft_toRightOf="@+id/user"
    app:layout_constraintBottom_toBottomOf="@+id/xueyuan"
    android:layout_marginTop="@dimen/normalDistance"
    android:layout_marginRight="@dimen/normalDistance"
    app:layout_constraintVertical_bias="0.481"
    app:layout_constraintRight_toRightOf="parent"
    android:layout_marginEnd="@dimen/normalDistance">
</Spinner >
```

（4） 实验遇到困难以及解决思路

Q1：不知道如何定义下拉框中的待选项。

解决方法：在查阅了相关的博客和资料后，发现要另外写一个 arrays.xml 文件，然后在 Spinner 中直接对 arrays 文件中的项进行引用。

六、 实验思考及感想

本次实验是设计一个 UI 的界面，总体实现较为简单，不过还是小坑不断。由于是第一次接触 Java 语言和 Android Studio 软件，所以对很多界面的操作和内置函数的熟悉度还不够，在对布局进行进一步设计时遇到了不小的麻烦，比如在两个文件里定义了相同的控件格式、给某个控件的属性多次赋了不

同的值等。

不过在这次实验中让我感触最深的还是 ConstraintLayout 和 LinearLayout 之间的区别与联系。因为刚开学，时间较为充裕，所以我分别用两种布局完成了本次实验。最开始的时候，我使用的是 LinearLayout，在进行最后两个按钮的整体居中时，我使用了 Layout 嵌套，即在整体的 LinearLayout 布局中又嵌套了一层 LinearLayout，虽然实现起来也没觉得有多复杂，但是从优化的角度上，减少层级的嵌套对 app 性能会有不小的帮助。所以从大工程的角度来看，ConstraintLayout 没有 layout 的嵌套，简洁了许多。同时通过直接在可视化界面进行操作可以降低编写复杂布局的难度，同时也调高了灵活度。

除此之外，我还在和同学的讨论中发现了解决问题的多种方法。比如最后两个 button 的打包问题，在与同学的讨论过程中我发现还可以把第一个按钮 button1 设置为 packed，然后两个按钮就会自动连接在了一起，之后调整 margin 也就很容易了。

最后，Android Studio 强大的自动补全功能真的满分。