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	15 级	专业（方向）	移动互联网
学号	15352071	姓名	戴斯铭
电话	13727023544	Email	908660116@qq.com
开始日期	2017.10.13	完成日期	2017.10.16

一、实验题目

事件处理。

二、实验目的

1.了解 Android 编程基础

2.熟悉 ImageView、Button、RadioButton 等基本控件，能够处理这些控件的基本事件

3.学会弹出基本的对话框，能够定制对话框中的内容，能对确定和取消按钮的事件做处理

三、实现内容

实现一个 Android 应用，界面呈现如下效果：

要求：

（1）该界面为应用启动后看到的第一个界面

（2）输入学号和密码的控件要求用 TextInputLayout 实现

（3）点击图片，弹出对话框如下图：

点击“拍摄”选项，弹出 Toast 信息“您选择了[拍摄]”；

点击“从相册选择”选项，弹出 Toast 信息“您选择了[从相册选择]；

点击“取消”按钮，弹出 Toast 信息“您选择了[取消]”。

（4）切换 RadioButton 的选项，弹出 Snackbar 提示“您选择了 xx”；

例如从选项“学生”切换到选项“教职工”，则提示“您选择了教职工”；

点击 Snackbar 上的“确定”按钮，则弹出 Toast 信息“Snackbar 的确定按钮被点击了”。

（5）点击登录按钮

依次判断学号是否为空，密码是否为空，用户名和密码是否正确（正确的学号和密码分别为“123456”，“6666”）；不正确则给出错误信息，如学号和密码都正确则提示“登陆成功”，如图：

（6）点击注册按钮

如果切换选项时，RadioButton 选中的是“学生”，那么弹出 Snackbar 信息“学生注册功能尚未启用”，如果选中的是“教职工”，那么弹出 Toast 信息“教职工注册功能尚未启用”。

四、课堂实验结果

（1）启动后的界面

实现过程：

此界面与实验一的界面基本相同，唯一的差别是将输入框进行了修改。因为我们需要使用 TextInputLayout 来实现学号和密码输入框的控件，因此，需要在 xml 代码中将两个 EditText 分别添加进两个 TextInputLayout 布局，再对 TextInputLayout 的 Constraint 等进行

设置即可。同时，删除原有的两个“学号”和“密码”TextView 控件。

其中一个 TextInputLayout 的代码如下：（另一个雷同，不贴代码）

```
<android.support.design.widget.TextInputLayout
    android:id="@+id/textinputlayout1"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginLeft="20dp"
    android:layout_marginRight="20dp"
    android:layout_marginTop="14dp"
    app:errorEnabled="true"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/imageView">

    <EditText...>
</android.support.design.widget.TextInputLayout>
```

在这个界面设置中，有两个需要提到的地方：

一个是焦点问题，由于界面在启动时自动将焦点设置在 xml 文件中找到的第一个 EditText 中，这会导致界面启动后焦点默认在学号输入框中，不符合要求。为了解决这一问题，我将界面一启动时的焦点放在了中大校徽的 ImageView 上，相关代码如下：

```
android:focusable="true"
android:focusableInTouchMode="true"
```

第二个是两个 EditText 的下划线粗细不一，从界面中也可以明显看出来。一开始我以为是自己的代码编写错误，但是两个 EditText 的 xml 代码基本一致，不存在这种可能。当我把项目放置在他人电脑上跑时，则可以正常显示。因此可能是 ROM 不同而导致的显示不同。

（2）在图片上使用对话框，并且实现相应的点击事件。实现界面如下：

点击拍摄后：

点击从相册选择：

点击取消后：

实现过程：

要在图片上点击后产生事件，即 ImageView 的点击事件 image. setOnClickListener()。在 java 文件中，首先需要将该 image 初始化，如下：

```
image = (ImageView)findViewById(R.id.imageView); //中大图片
```

之后再调用 setOnClickListener 来产生事件。而在写其内容之前，首先需要实现对话框。在本次实验中，由于需要使用按钮和列表，因此选用 AlertDialog 来实现。首先，需要调用 AlertDialog.Builder 来初始化，如下：

```
final AlertDialog.Builder alertDialog = new AlertDialog.Builder(this);
```

之后再调用相应的方法即可。比如，需要设置标题“上传头像”，则使用 setTitle()；需要设置可选项的列表，则使用 setItems()；需要设置取消按钮，则使用 setNegativeButton()方法。而在列表实现中，同样需要设置点击事件，在 setItems()中的参数为 string 数组，因此可以添加数组，将“拍照”和“从相册选择”添加进去。而在区分点击事件时，则可通过判断点击的是哪个数组下标来区分，相应代码如下：

```
alertDialog.setTitle("上传头像").setItems(new String[] {"拍摄","从相册选择"},
        new DialogInterface.OnClickListener(){
            @Override
            public void onClick(DialogInterface dialogInterface, int i){
                if(i == 0){
                    Toast.makeText(getApplicationContext(),"您选择了[拍摄]",Toast.LENGTH_SHORT).show();
                }
                if(i == 1){
                    Toast.makeText(getApplicationContext(),"您选择了[从相册选择]",Toast.LENGTH_SHORT).show();
                }
            }
```

由上可以看到，点击后产生的事件是弹出 Toast 信息。Toast 事件的产生十分简单，只需要调用 makeText 方法即可。三个参数分别为上下文，要显示的信息以及时长。并且需要注意的是，最后需要写上 show()，这样才能使其在应用中显示出来。

同理，设置“取消”按钮也需要同时有点击事件，同样弹出 Toast 信息。相关代码如下：

```
}).setNegativeButton("取消",
new DialogInterface.OnClickListener(){
    @Override
    public void onClick(DialogInterface dialogInterface, int i){
        Toast.makeText(MainActivity.this,"你选择了取消",Toast.LENGTH_SHORT).show();
    }
}).create();
```

最后，添加 create()函数，则 alertDialog 的设置完成。之后，再在图片的点击事件中直接使用它即可，实现如下：

```
if(image != null){
    image.setOnClickListener(new View.OnClickListener(){
        @Override
        public void onClick(View view){
            alertDialog.show();
        }
    });
}
```

（3）切换 RadioButton 的选项，弹出 Snackbar 提示“您选择了 xx”。实现界面如下：

实现过程：首先，需要实现 RadioGroup 的切换事件，就要用 RadioGroup 的 setOnCheckedChangeListener()方法。当然，需要引入相关的控件，如下：

```
person = (RadioGroup)findViewById(R.id.radioGroup);
student = (RadioButton)findViewById(R.id.radioButton4);
staff = (RadioButton)findViewById(R.id.radioButton5);
```

随后，再在 person.setOnCheckedChangeListener()方法中做处理。在里面重写函数，如下：

```
person.setOnCheckedChangeListener(new RadioGroup.OnCheckedChangeListener(){
    @Override
    public void onCheckedChanged(RadioGroup group, int checkedId){
```

通过 onCheckedChanged 函数的参数也可以看到，通过 checkedId 参数即可获取按钮的切换，从而实现不同的点击事件。实验要求我们，在进行选项切换时，弹出 Snackbar 提示选择，因此需要实现 Snackbar。下面以选项切换到“学生”时为例。

Snackbar 的创建需要使用 make 方法，其内的三个参数分别为根布局、要显示的消息以及时长。因此我实现如下：

```
if(checkedId == student.getId()){
    Snackbar snackbar = Snackbar.make(group, "您选择了学生",Snackbar.LENGTH_LONG);
```

除此之外，还要再 Snackbar 上设置一个确定按钮，当点击其时弹出 Toast 信息“Snackbar 的确定按钮被点击了”。设置确定按钮可以调用 Snackbar 的方法 setAction，在里面调用 OnClickListener 设置点击事件即可。需要弹出 Toast 信息，方法与之前类似，不再赘述。实现如下：

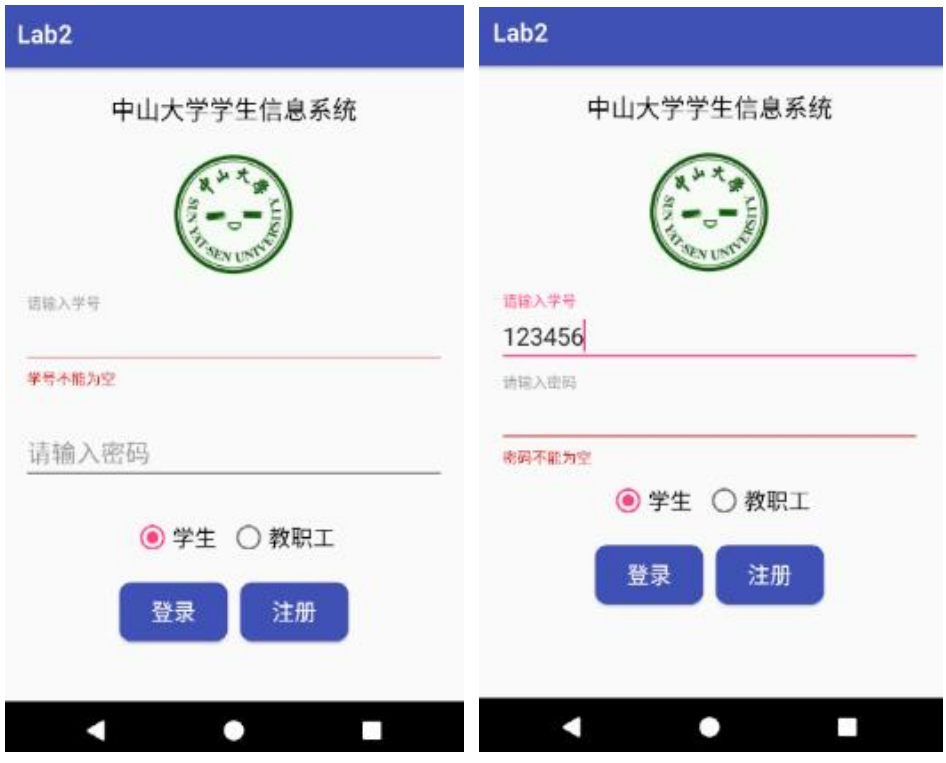
```
snackbar.setAction("确定", new View.OnClickListener(){
    @Override
    public void onClick(View view){
        Toast.makeText(MainActivity.this,"Snackbar的确定按钮被点击了",Toast.LENGTH_SHORT).show();
    }
}).show();
```

需要特别注意的是，要使效果显示在应用上，需要记得添加 show()。之后，由于在 Snackbar 上的按钮默认为桃红色，不太好看，因此我将其颜色修改为天蓝色：

```
snackbar.setActionTextColor(getResources().getColor(R.color.skyblue));
```

切换至教职工选项的写法与切换至学生相同，不再赘述。

（4）点击登录按钮，实现界面如下：



（与上一内容类似，点击确定按钮后会弹出 Toast 信息，不再显示）

实现过程：这一部分的内容需要我们充分利用 TextInputLayout 的功能。一般情况下，在 EditText 里输入内容时，原有的提示会消失，而将 EditText 加入 TextInputLayout 后则可以使 hint 成为一个浮动的标签。

当然，首先需要将 TextInputLayout 引入 java 文件，同时通过 TextInputLayout 获取 EditText 中的内容。

```
user_idText = (TextInputLayout)findViewById(R.id.textinputlayout1);
passwordText = (TextInputLayout)findViewById(R.id.textinputlayout2);
mNumberEdit = user_idText.getEditText();
mPassEdit = passwordText.getEditText();
```

随后，再调用 TextInputLayout 的 setHint 方法，即可设置浮动的提示。如下：

```
user_idText.setHint("请输入学号");
passwordText.setHint("请输入密码");
```

除了有浮动的标签之外，TextInputLayout 还设有报错功能。通过依次判断学号是否为空，密码是否为空来在输入框下方弹出错误提示。因此我使用 if 语句进行判断，当学号框输入为空时，将 setErrorEnabled 设置为 true，再调用 setError("学号不能为空")；当不为空时，则将 setErrorEnabled 设置为 false。密码框的设置同理，实现如下：

```
if(TextUtils.isEmpty(user_id)){
    user_idText.setErrorEnabled(true);
    user_idText.setError("学号不能为空");
    return;
} else{
    user_idText.setErrorEnabled(false);
}
if(TextUtils.isEmpty(password)){
    passwordText.setErrorEnabled(true);
    passwordText.setError("密码不能为空");
} else{
    passwordText.setErrorEnabled(false);
}
```

当然，以上的内容需要填写在登录按钮的点击事件中。当两个框内的内容不为空时，此时需要判断其所填信息是否正确。正确的学号和密码分别为“123456”，“6666”，若正确则弹出 Snackbar 信息显示登录成功，否则则显示学号或密码错误。同样的，使用 if 语句来进行判断即可，将从 EditText 中获取的两个字符串分别与“123456”和“6666”比较，之后再实现 Snackbar 内容即可，如下：

```
if(user_id.equals("123456") && password.equals("6666")){
    Snackbar snackbar = Snackbar.make(login, "登录成功", Snackbar.LENGTH_LONG);
    snackbar.setAction("确定",(view) → {
            Toast.makeText(MainActivity.this, "Snackbar的确定按钮被点击了",Toast.LENGTH_SHORT).show();
    }).show();
    snackbar.setActionTextColor(getResources().getColor(R.color.skyblue));
}
else if((!user_id.equals("123456") || !password.equals("6666")) &&
        (!TextUtils.isEmpty(user_id)&&!TextUtils.isEmpty(password))){
    Snackbar snackbar = Snackbar.make(login, "学号或密码错误", Snackbar.LENGTH_LONG);
    snackbar.setAction("确定",(view) → {
            Toast.makeText(MainActivity.this, "Snackbar的确定按钮被点击了",Toast.LENGTH_SHORT).show();
    }).show();
    snackbar.setActionTextColor(getResources().getColor(R.color.skyblue));
}
```

同样的，Snackbar 的实现与之前相同，不再赘述。

（5）点击注册按钮，实现界面如下：

实现过程：实验要求，当点击学生注册后弹出 Snackbar 消息，显示“学生注册功能尚未启用”；当点击教职工注册后弹出 Toast 信息，显示“教职工注册功能尚未启用”。因此，需要设置对注册按钮的点击事件，同时，还需要从 radioGroup 中获取选中的 Button 的 Id。

```
register = (Button) findViewById(R.id.button2);
register.setOnClickListener(new View.OnClickListener(){
    @Override
    public void onClick(View view){
        RadioGroup radioGroup = (RadioGroup) findViewById(R.id.radioGroup);
        int checkedId = radioGroup.getCheckedRadioButtonId();
```

从图中可以看到，为了得到 RadioGroup 的对应 Id，需要获取用户的选择点击。随后，再根据 checkedId 来实现是显示 Snackbar 消息还是 Toast 消息就好了。当选中的是学生时，实现如下：

```
if(checkedId == R.id.radioButton4){
    Snackbar snackbar = Snackbar.make(register, "学生注册功能尚未启用",Snackbar.LENGTH_LONG);
    snackbar.setAction("确定", (view) → {
            Toast.makeText(MainActivity.this,"Snackbar的确定按钮被点击了",Toast.LENGTH_SHORT).show();
    }).show();
    snackbar.setActionTextColor(getResources().getColor(R.color.skyblue));
}
```

同样的，点击 Snackbar 上的确定按钮会弹出 Toast 消息，而当选中的是教职工时，实现如下：

```
if(checkedId == R.id.radioButton5){
    Toast.makeText(MainActivity.this,"教职工注册功能尚未启用",Toast.LENGTH_SHORT).show();
}
```

至此，所有基本功能实现完毕。

五、课后实验结果

课上老师有介绍过自动完成文本控件 AutoCompleteTextView，而这次实验中使用了两个 EditText，AutoCompleteTextView 是 EditText 的子类。因此，我将原有的学号输入框，从 EditText 替换成了 AutoCompleteTextView。

AutoCompleteTextView 与 EditText 的功能基本相同，唯一的特点是增加了自动完成文本的功能。首先，在 xml 文件里将 EditText 替换成 AutoCompleteTextView 控件，如下：

```
<AutoCompleteTextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/editTextUser_id"
    android:layout_marginLeft="0dp"
    android:layout_marginRight="0dp"
    android:layout_marginTop="13dp"
    android:inputType="number"
    android:textSize = "20sp"
    android:completionThreshold="2"
    app:layout_constraintLeft_toRightOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/imageView"/>
```

从代码中的 android:completionThreshold=“2”这一行可以看到，我将学号的阈值设置为 2，当输入两个数字后，会有弹出与输入信息接近的提示信息，即相对应的学号供我们选择。而这个控件需要在 java 中添加相应的代码：

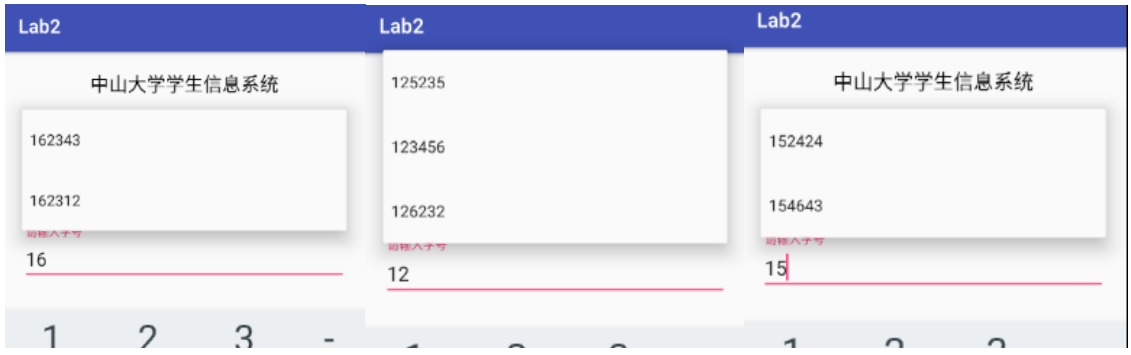
首先，设置字符数组，用于匹配：

```
String[] id_number = {"125235","162343","123456","126232","174521",
                      "162312","152424","154643","142452"};
```

接着，建立 ArrayAdapter 类，将 AutocompleteTextView 与 string 数组联系起来，并设置提示信息的样式；然后调用 AutocompleteTextView 的 setAdapter 方法将 adapter 指定给 mNumberEdit。

```
ArrayAdapter<String> adapter = new ArrayAdapter<~>(MainActivity.this,
        android.R.layout.simple_dropdown_item_1line,id_number);
mNumberEdit.setAdapter(adapter);
```

其余的功能与 EditText 完全相同，最终实现如下截图：



（注：上交的代码文件是此修改版本。）

六、实验思考及感想

这次实验花了较长的时间完成，一是对各种控件引入 java 文件的不熟悉，二是对各种方法调用的不了解。同时，我在撰写实验报告时，也发现了自己的一个大缺点，就是在设置各个控件的 id 没有特别的区分。从上一次实验开始，我在 xml 中的控件 id 都是其自动生成的，没有多加理会，但是在这次实验中，则明显感觉到了这一写法的不足。按钮那么多，如果没有都是以 button+数字的方式来命名，那么根本就很难区分，因此下次我会对其进行修改。

同时，这次实验也提高了自学能力吧，在实验课上的电脑只能看到一幅动态彩色的马赛克图像，所以 TA 讲了什么怎么讲的也基本看不到，所以都是自己在网上各种搜索控件的使用方法。虽然过程有点痛苦，但是在看到成果时，也是感到很开心的。我觉得，这也是安卓开发或者 iOS 开发的最吸引人的地方！