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	专业（方向）	移动互联网
学号	15352163	姓名	李德明
电话	15352409815	Email	1335188820@qq.com
开始日期	10.13	完成日期	10.16

一、实验题目

实验二 事件处理

二、实现内容

实现一个Android应用，界面呈现与实验一基本一致，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，
点击“拍摄”，弹出Toast信息“您选择了[拍摄]”；
点击“从相册选择”，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”；
点击Snackbar的确定按钮，则弹出Toast“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”和“6666”），不正确则给出错误信息，如学号和密码都正确则提示“登录成功”。

（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Snackbar信息“教职工注册功能尚未启用”。

三、课堂实验结果

（1）实验截图


2:56
Lab2
中山大学学生信息系统
请输入学号
请输入密码
学生
教职工
登录
注册

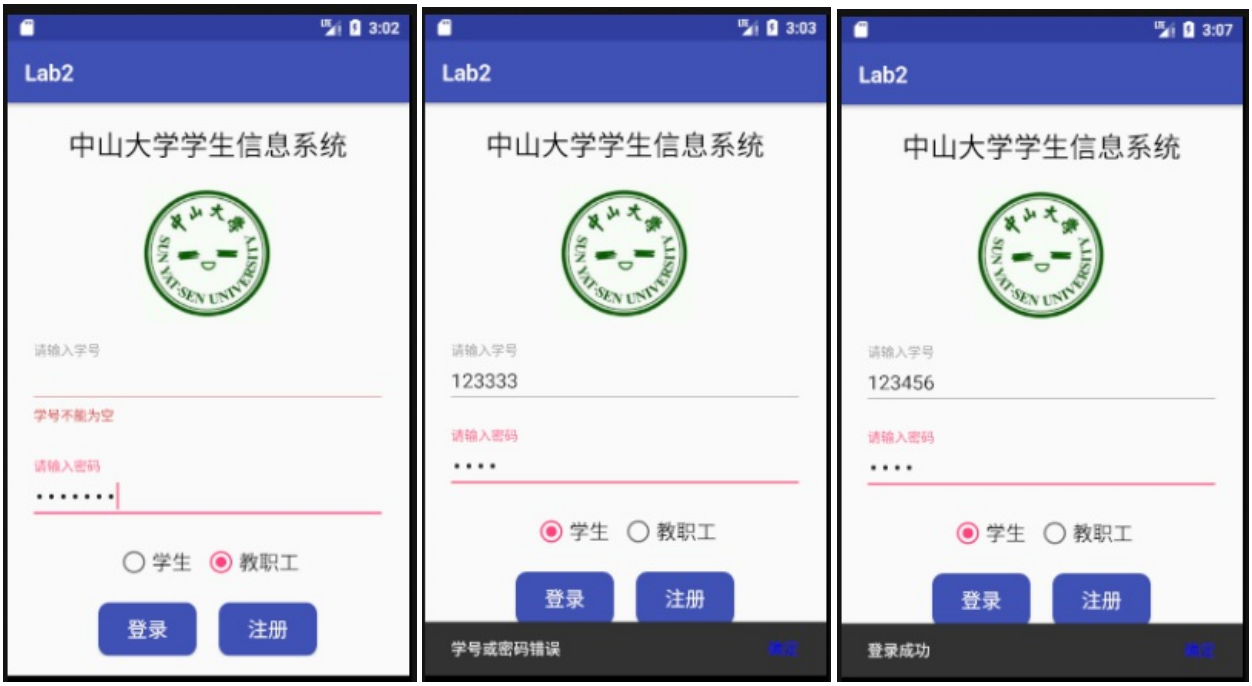

2:57
Lab2
中山大学学生信息系统
上传头像
拍摄
从相册选择
取消
请输入密码
学生
教职工
登录
注册


2:59
Lab2
中山大学学生信息系统
请输入学号
请输入密码
学生
教职工
您选择了[从相册选择]


3:00
Lab2
中山大学学生信息系统
请输入学号
请输入密码
学生
教职工
登录
注册
您选择了教职工


3:01
Lab2
中山大学学生信息系统
请输入学号
请输入密码
学生
教职工
登录
注册
教职工注册功能尚未启用


3:01
Lab2
中山大学学生信息系统
请输入学号
请输入密码
学生
教职工
Snackbar的确定按钮被点击了



（2）实验步骤以及关键代码

（1）首先，对于约束文件，我们可以在实验一的基础上进行修改，在本次实验中我将实验一中的约束布局改成了线性布局，具体代码可见工程源码Lab2。实验要求输入学号和密码的两个EditText需要用TextInputLayout实现，这里要注意的是一个TextInputLayout只能包含一个EditText。下面是EditText部分代码：

```
<android.support.design.widget.TextInputLayout
    android:id="@+id/number"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:layout_marginLeft="20dp"
    android:layout_marginRight="20dp">
    <EditText
        android:id="@+id/num"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="请输入学号"
        android:textSize="18sp"
        android:inputType="number"
        />
</android.support.design.widget.TextInputLayout>
```

要实现该布局，我们要在build.gradle文件中加入如下代码：

```
compile 'com.android.support:design:26.0.0-alpha1'
```

用TextInputLayout的好处是可以实时监控输入内容，对输入内容报出相应的错误。

（2）接下来我们来实现MainActivity.java文件。

a). 首先我们实现点击图片显示的对话框的完整功能。先创建一个ImageView的对象mImage，然后把它和约束文件中的图片部件通过id关联起来。然后创建这个对象的点击监听器。

在实现mImage的OnClick函数的时候，我们要实现一个对话框（AlterDialog）,它的标题为“上传头像”，两个下拉列表项为“拍摄”和“从相册选择”，还有一个NegativeButton“取消”。对于两个下拉列表项的onClick函数，弹出Toast，获得它们的名称，然后显示“您选择了xxx”；对于NegativeButton，它的onClick函数里面直接弹出toast“您选择了[取消]”。

代码如下：

```
ImageView mImage=(ImageView) findViewById(R.id.sysu);
mImage.setOnClickListener(new View.OnClickListener()
{
    @Override
    public void onClick(View v)
    {
        AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);
        builder.setTitle("上传头像");//   title
        final String[] method = {"拍摄", "从相册选择"};//    choose item
        builder.setItems(method, new DialogInterface.OnClickListener()
        {
            @Override
            public void onClick(DialogInterface dialog, int which)
            {
                Toast.makeText(MainActivity.this, "您选择了[" + method[which]+"]", Toast.LENGTH_LONG).show();
            }
        });
        builder.setNegativeButton("取消", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                Toast.makeText(MainActivity.this, "您选择了[取消]", Toast.LENGTH_SHORT).show();
            }
        });
        builder.show();
    }
});
```

b). 接下来实现两个RadioButton切换的效果。首先创建一个RadioGroup的对象mRadioGroup，然后把它和约束文件中的单选按钮组关联起来。通过设置一个切换监视器来实现按钮切换时弹出的信息。

这里要用到Sanckbar来弹出信息。当按钮切换时，获得选中按钮的名称，然后弹出“您选择了xx”。这里关键是onCheckedChanged函数的实现，它传进了两个参数：一个是单选按钮组，另一个是被选中的按钮的id。实现Snackbar.make()函数时，要注意将Snackbar弹出时的“确定”按钮的颜色变为实验要求的蓝色，用.setActionTextColor()函数实现即可。

代码如下：

```
final RadioGroup mRadioGroup=(RadioGroup) findViewById(R.id.radio);
mRadioGroup.setOnCheckedChangeListener(new RadioGroup.OnCheckedChangeListener() {
    @Override
    public void onCheckedChanged(RadioGroup group, int checkedId) {
        RadioButton checkedone=(RadioButton) findViewById(checkedId);
        Snackbar msnackbar1=Snackbar.make(findViewById(R.id.radio), "您选择了"+checkedone.getText().toString(), Snackbar.LENGTH_LON
        msnackbar1.setAction("确定", new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Toast.makeText(MainActivity.this, "Snackbar的确定按钮被点击了", Toast.LENGTH_SHORT).show();
            }

        }).setActionTextColor(Color.BLUE).show();
    }
});
```

c). 实现注册按钮的监听。这里要注意的是要实现这个模块，要用到RadioGroup的切换监听。那么我们的思路就很清晰了：首先设置注册按钮的监听器，然后在onClick函数里面实现嵌套一个RadioGroup的切换监听器。这样一来，当监听到注册按钮被点击时，先通过RadioGroup的切换监听器找到当前选中的单选按钮的名称，然后执行用一个Snackbar弹出“xx注册功能尚未启用”。

代码如下：

```
Button btn2=(Button) findViewById(R.id.register);
final RadioGroup radioSelect=(RadioGroup) findViewById(R.id.radio);
btn2.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        int checkedId=radioSelect.getCheckedRadioButtonId();
        RadioButton checkButton=(RadioButton) findViewById(checkedId);
```

```
        Snackbar msnackbar2=Snackbar.make(findViewById(R.id.register), checkButton.getText().toString()+"注册功能尚未启用", Snackbar.
        msnackbar2.setAction("确定", new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Toast.makeText(MainActivity.this, "Snackbar的确定按钮被点击了", Toast.LENGTH_SHORT).show();
            }

        }).setActionTextColor(Color.BLUE).show();
    }
});
```

d). 实现登录按钮。

这部分是最复杂的，首先我们来分好层次：顶层是登录按钮的点击监听器，然后在它的onClick函数里面开始实现下一层的操作。

这里所述的下一层的具体步骤为：

首先判断学号是否为空，用isEmpty函数来判断。如果为空，则TextInputLayout类的对象t1也就是输入学号的EditText就会报错为“学号不能为空”。这里要注意一个细节。如果判断不为空，则取消报错，用setErrorEnable(false)实现。如果不加这部分，只要报错出现就不会消失。

然后判断密码是否为空。判断方法和学号的判断方法一样。还有一个细节是：如果学号和密码都为空，然后点击登录时，只有学号的控件报错！实现方法可以在下面代码看到。

第三步判断输入的学号和密码是否和实验文档要求的一样，这里不能用“==”，要用.equals函数。

如果相等，则用Snackbar弹出提示登录成功的信息；

若两个都不为空，但是输入的学号和密码有误，则用Snackbar弹出“学号或密码错误”的信息；

如果其中有一个为空或者两个都为空，则Snackbar不执行，它会在上面TextInputLayout的报错里面执行的。

```
Button btn=(Button) findViewById(R.id.login);
final TextInputLayout t1=(TextInputLayout) findViewById(R.id.number);
final TextInputLayout t2=(TextInputLayout) findViewById(R.id.password);
final EditText username= (EditText) findViewById(R.id.num);
final EditText passowrd= (EditText) findViewById(R.id.pas);

btn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
            if(TextUtils.isEmpty(username.getText().toString())) {
                t1.setError("学号不能为空");
            }
            else {
                t1.setErrorEnabled(false);
            }

            if(TextUtils.isEmpty(passowrd.getText().toString())) {
                if(TextUtils.isEmpty(username.getText().toString())) {
                    t2.setErrorEnabled(false);
                }
                else {
                    t2.setError("密码不能为空");
                }
            }
            else {
                t2.setErrorEnabled(false);
            }

        if(username.getText().toString().equals("123456") && passowrd.getText().toString().equals("6666")) {
            Snackbar msnackbar3=Snackbar.make(findViewById(R.id.login), "登录成功", Snackbar.LENGTH_LONG);
            msnackbar3.setAction("确定", new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    Toast.makeText(MainActivity.this, "Snackbar的确定按钮被点击了", Toast.LENGTH_SHORT).show();
                }
```

```
            }).setActionTextColor(Color.BLUE).show();
        }
        else if(username.getText().toString().equals("") || passowrd.getText().toString().equals("")) {

        }
        else {
            Snackbar msnackbar4=Snackbar.make(findViewById(R.id.login), "学号或密码错误", Snackbar.LENGTH_LONG);
            msnackbar4.setAction("确定", new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    Toast.makeText(MainActivity.this, "Snackbar的确定按钮被点击了", Toast.LENGTH_SHORT).show();
                }

            }).setActionTextColor(Color.BLUE).show();
        }
    }
});
```

（3）实验遇到的困难以及解决思路

遇到的第一个问题是TextInputLayout的使用，因为忽略了在gradle文件中添加所支持的库的版本，导致实现的时候一直报错，这是我的个人疏忽，后来看了实验文档，添加了库之后就好了。

第二个问题是Snackbar的用法。由于之前上课时候并没有过多的讲解该组件，经过查找资料，我发现Snackbar不仅比Toast更加好用，而且可扩展性更好，还可以随意修改样式。这种组件是目前软件开发中常用的组件。

在实现注册按钮的时候，一开始的时候是想直接用上面实现的切换监听器来实现，后来发现实现之后当单选按钮切换的时候他可以直接绕过注册按钮的监听器直接显示Snackbar。然后我发现其实不用在切换按钮监听器里面实现的，直接在下面实现，只需要增加一个RadioGroup的对象就可以了，要获得被选中按钮的名称的时候调用checkButton.getText().toString()就可以了。这里我们还用到了之前切换监视函数里面的CheckedId变量。

在最后实现登录按钮的时候，最初实现的demo里面如果学号和密码都不输入，然后直接点击登录的时候，两个输入框都会报错，而实验debug.apk里面给出的样例是只显示学号输入框的报错。后来在判断的时候加了一个判断“如果学号输入框为空，那么密码输入框禁止报错”，这样，只要学号是空的，密码输入框就不会报错，问题就得到解决了。

四、课后实验结果

增加了学号长度限制报错以及密码可见按钮。

对于添加密码可见按钮，我们只需要在xml文件密码输入框对应的TextInputLayout设置中加上以下代码就可以了。

```
app:passwordToggleEnabled="true"
```

对于添加学号长度限制，我们首先要在xml文件学号输入框对应的TextInputLayout设置中加上以下代码：

```
app:counterMaxLength="6"
app:counterEnabled="true"
```

上面代码的意思是：设置输入框最大输入长度为6，然后增加计数器。

接下来，我们在MainActivity文件中添加如下代码：

```
username.addTextChangedListener(new TextWatcher() {
    @Override
    public void beforeTextChanged(CharSequence s, int start, int count, int after) {
    }
    @Override
    public void onTextChanged(CharSequence s, int start, int before, int count) {
    }
    @Override
    public void afterTextChanged(Editable s) {
        if(username.getText().length() > t1.getCounterMaxLength()) {
            t1.setError("学号长度超过限制");
        }
    }
});
```

上面代码是给学号输入框增加了一个文本改变监视器，当输入文本改变前、中、后都可以执行相应的指令。在这里我们只实现文本改变后的监视。

五、实验思考及感想

本次实验是第一次真正意义上的软件开发，上次实验只是简单的添加约束文件，然后改相应的变量就可以了，而这次的实验要求实现基本事件处理，这就要用到很多java的知识。因为我们没有系统的学过java语言，即使有C++的基础也显得很吃力，有很多语句要好长时间才能理解。本次实验也花了大量的时间和精力在上面，查了很多资料，不断试错，终于理解了实现的原理。

搞清楚原理之后，我用一个很高效的方法：画流程图。先将思路和软件执行时候的流程画出来，然后照着流程写代码，效率会高很多。

通过本次实验，我发现安卓开发的时候可以用很多库，但是我们队库的了解还不是很详细，甚至有些库我们都不知道它的存在，所以我要在这方面下点苦功，有了轮子拿来用就可以，为什么要费力去再造一个轮子出来呢。

总结一席本次实验，前半段完全是在解读实验文档以及查资料和语法，后半段出错频频，很多细节上的问题出现bug，不过最终都改过来了，在实现拓展功能的时候就更加得心应手了。

本次实验的另外一个拓展项就是我把布局形式从实验一的约束布局换成了现在的线性布局。发现线性布局在这种小程序上要比约束布局简单明了很多！

本次实验收获颇丰，也算是正式踏上了java的自学之路，加油吧！