中山大学数据科学与计算机学院软件工程（移动信息工程）本科生实验报告

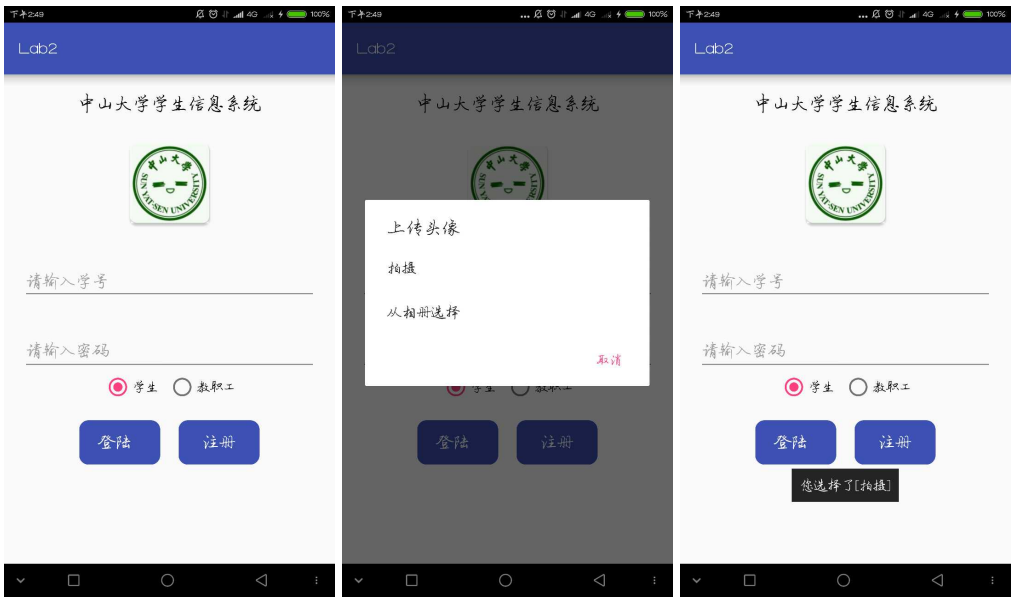
# 移动应用开发实验报告

| 姓名 | 学号 | 班级 | 电话 | 邮箱 | 日期 |
| --- | --- | --- | --- | --- | --- |
| 蔡烨 | 15352010 | 1501班 | 15989010669 | caiy29@mail2.sysu.edu.cn | 2017/10/15 |

## 1. 实验题目

实现一个Android应用，界面如下图，且能够解决基本的控件触发事件，弹出对话框、弹出toast、弹出snackbar，判断输入是否正确等。


## 2. 实现内容




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
登陆
注册
您选择了[从相册上传]


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
登陆
注册
您选择了[取消]


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
登陆
注册
您选择了教职工
确定


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
登陆
注册
Snackbar 的确定按钮被点击了

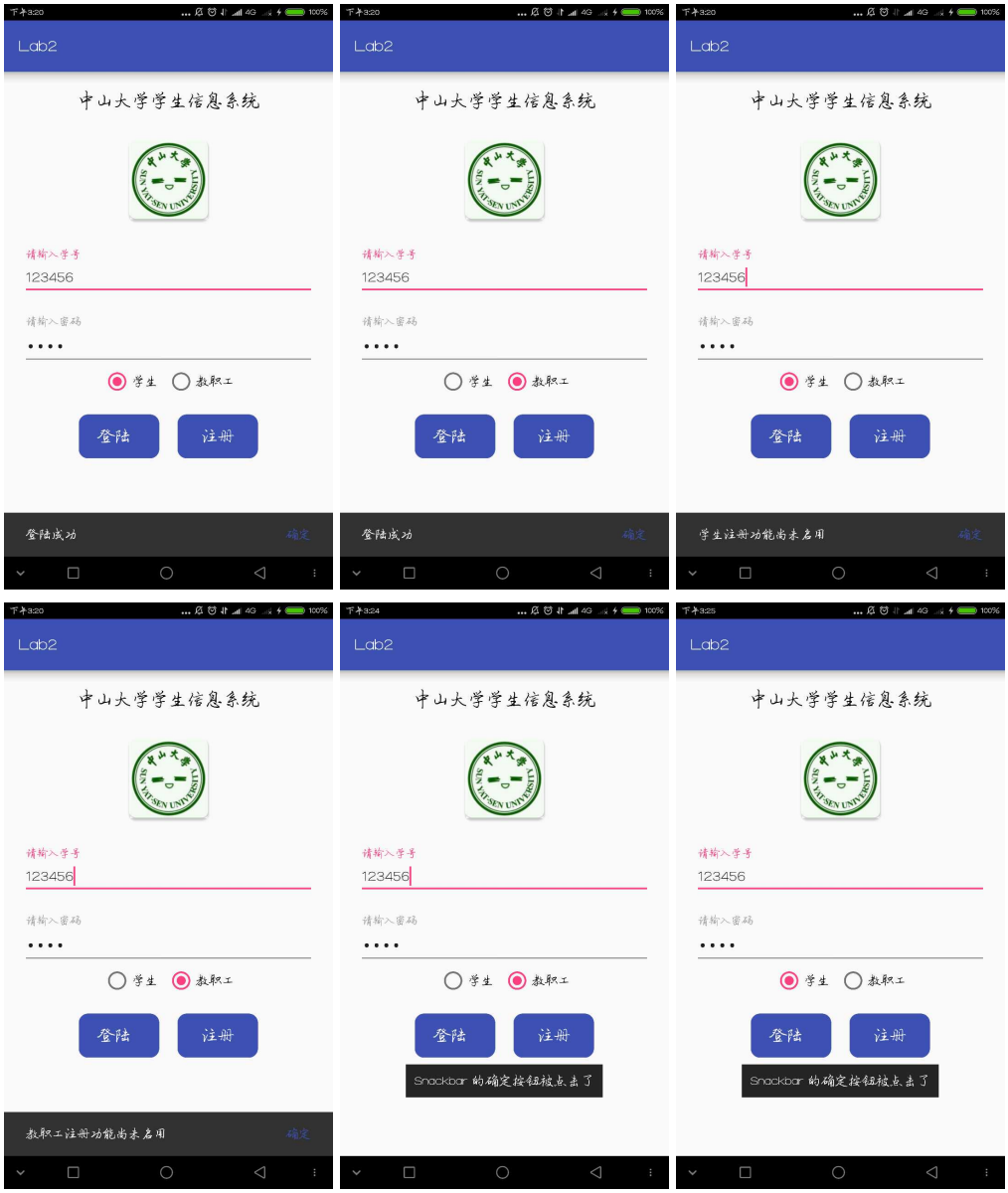

Lab2
中山大学学生信息系统
请输入学号
请输入密码
学生
教职工
登陆
注册
您选择了学生
确定


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
登陆
注册
Snackbar 的确定按钮被点击了


Lab2
中山大学学生信息系统
请输入学号
学号不能为空
请输入密码
学生
教职工
登陆
注册


Lab2
中山大学学生信息系统
请输入学号
12345
请输入密码
密码不能为空
学生
教职工
登陆
注册


Lab2
中山大学学生信息系统
请输入学号
12345
请输入密码
学生
教职工
登陆
注册
学号或密码错误
确定




## 3. 实验过程

### 1. 应用打开的第一个界面如第一个图

在控件处加上获取焦点的代码，打开应用时，光标（焦点）就默认在那个控件上。此处应把焦点放在整个屏幕上。

```
<android.support.constraint.ConstraintLayout xmlns:android="http
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.caiye.lab2.MainActivity"
    android:focusable="true"
    android:focusableInTouchMode="true">
```

### 2. 用**TextInputLayout**实现输入学号和密码的控件

- 首先修改xml文件中的内容，删去lab1中的“学号”和“密码”两个TextView，将剩下的两个EditText分别包裹在两个TextInputLayout中。此处需要注意的是，这样一改动会破坏原先的依赖关系，需要将原本关于两个EditText的依赖关系分别挪到对应的TextInputLayout中，并修改依赖关系中原先对应的EditText的id为TextInputLayout的id。也就是说，将TextInputLayout包裹着EditText作为一个

整体来布局，将EditText的高度和宽度都设置为match_parent，而TextInputLayout的高度则为wrap_content，宽度为0dp（为了让该控件自动填充空间）。

```
<android.support.design.widget.TextInputLayout
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:id="@+id/sid"
    android:onClick="onClick"
    android:layout_marginLeft="20dp"
    android:layout_marginRight="20dp"
    app:layout_constraintBottom_toTopOf="@+id/password"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/image"
    android:layout_marginStart="20dp"
    android:layout_marginEnd="20dp">

    <EditText
        android:id="@+id/input1"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:hint="请输入学号"
        android:inputType="number"
        android:textSize="18sp" />
</android.support.design.widget.TextInputLayout>
```

```
<android.support.design.widget.TextInputLayout
    android:layout_height="wrap_content"
    android:layout_width="0dp"
    android:id="@+id/password"
    android:onClick="onClick"
    android:layout_marginTop="20dp"
    android:layout_marginRight="20dp"
    android:layout_marginLeft="20dp"
    app:layout_constraintTop_toBottomOf="@+id/sid"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintLeft_toLeftOf="parent"
    android:layout_marginStart="20dp"
    android:layout_marginEnd="20dp">

    <EditText
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/input2"
        android:hint="请输入密码"
        android:inputType="textPassword"
        android:textSize="18sp" />
</android.support.design.widget.TextInputLayout>
```

- 然后修改build.gradle文件中的配置，添加语句：compile 'com.android.support:design:25.3.1'。此处的版本号25.3.1应该与原来文件中的compile 'com.android.support:appcompat-v7:25.3.1'中的版本号对应。然后点击Sync Now同步项目。

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2', {
        exclude group: 'com.android.support', module: 'support-annotations'
    })
    compile 'com.android.support:appcompat-v7:25.3.1'
    compile 'com.android.support.constraint:constraint-layout:1.0.0-alpha7'
    testCompile 'junit:junit:4.12'

    compile 'com.android.support:design:25.3.1'
}
```

3. 点击图片，弹出对话框，点击对话框的不同选项，弹出不同的**toast**提示信息

- 在MainActivity.java文件中onCreate(Bundle savedInstanceState)函数里面，在setContentView(R.layout.activity_main);加载完布局之后，添加相应的代码。
- 首先用findViewById()函数并且转化为(ImageView)后得到图片控件，然后为控件绑定监听器。一旦图片被点击，便创建一个 AlertDialog.Builder对象，然后用setTitle()方法设置标题，用setItems()设置两个选项，用setNegativeButton()方法设置“取消”按钮，最后用show()方法展示出对话框。

```
ImageView mImage = (ImageView) findViewById(R.id.image);
mImage.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view){
        AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);
        builder.setTitle("上传头像");
        String[] options = {"拍摄","从相册选择"};
        builder.setItems(options,selectListener);
        builder.setNegativeButton("取消",new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialogInterface, int i) {
                Toast.makeText(MainActivity.this,"您选择了[取消]",Toast.LENGTH_SHORT).show();
            }
        });
        builder.show();
    }
```

- 设置选项时，绑定了自定义的DialogInterface.OnClickListener类型的函数selectListener，一旦选项被点击，调用Toast.makeText().show()函数弹出toast消息。因为选项是以一个string数组传进去的，用各自的下标来区分被点击的是哪一个选项。

```
final DialogInterface.OnClickListener selectListener = new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialogInterface, int i) {
        switch (i){
            case 0:
                Toast.makeText(MainActivity.this,"您选择了[拍摄]",Toast.LENGTH_SHORT).show();
                break;
            case 1:
                Toast.makeText(MainActivity.this,"您选择了[从相册上传]",Toast.LENGTH_SHORT).show();
                break;
            default:
                break;
        }
    }
};
```

- 设置取消按钮的时候，绑定了一个匿名函数对其进行监听，一旦被点击，用Toast.makeText().show()函数弹出toast消息。

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

4. 切换 **RadioButton** 的选项，弹出 **Snackbar** 提示**“您选择了 xx”**

- 首先用findViewById()分别获取RadioGroup和两个RadioButton控件，然后用setOnCheckedChangeListener()函数为RadioGroup注册监听器。

```
final RadioGroup radioGroup = (RadioGroup) findViewById(R.id.occasion);
final RadioButton student = (RadioButton)findViewById(R.id.student);
final RadioButton teacher = (RadioButton)findViewById(R.id.teacher);
radioGroup.setOnCheckedChangeListener(new RadioGroup.OnCheckedChangeListener() {
    @Override
    public void onCheckedChanged (RadioGroup group, int checkedId) {
```

- 其次要区别被点击的是哪一个RadioButton，onClick()函数中传入的checkedId是int类型的，代表是的被点击到的button的id的序号，而不是id本身(?)。而控件的getId()函数，返回是也是这个序号，而不是id本身，因此可以直接用两者进行对比，以此判断出被点击的是哪一个button。

```
public void onCheckedChanged (RadioGroup group, int checkedId) {
    String message = "";
    if(checkedId == student.getId())
        message = "您选择了学生";
    else if(checkedId == teacher.getId())
        message = "您选择了教职工";
```

- 调用Snackbar.make()函数，该函数传进去三个参数，分别是根布局，要显示的消息和时长。要显示的消息，在判断被点击的是哪一个控件时给了不同的赋值。而根布局在一开始就使用函数((ViewGroup)findViewById(android.R.id.content)).getChildAt(0)获取。之后用setAction()方法设定snackbar上的按钮，包括为按钮注册匿名函数的监听器。用setActionTextColor()方法设置按钮的颜色，用setDuration()设置snackbar弹出的时长，单位为ms。最后调用show()方法。

```
final View rootView = ((ViewGroup)findViewById(android.R.id.content)).getChildAt(0);

Snackbar.make(rootView,message,Snackbar.LENGTH_SHORT)
        .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();
            }
        })
        .setActionTextColor(getResources().getColor(R.color.colorPrimary))
        .setDuration(3000)
        .show();
```

- 实际上，snackbar.make()函数中第一个参数，传入的是当前控件，make函数里面使用了方法用来对当前控件一层一层往底层找去，使用找到的非空的底层view。因为我们在xml中使用的是constraintLayout，并没有coordinateLayout，因此最后使用的是便是id为android.R.id.content的根布局。因此此处可以直接将根布局传进去，也可以将当前控件传进去。

进入make方法看一下：

```
public static Snackbar make(@NonNull View view, @NonNull CharSequence text,
        @Duration int duration) {
    Snackbar snackbar = new Snackbar(findSuitableParent(view));
    snackbar.setText(text);
    snackbar.setDuration(duration);
    return snackbar;
}
```

其实这里面的重点就是Snackbarsnackbar =newSnackbar(findSuitableParent(view)); 我们可以看到我们传入的view经过了findSuitableParent()方法的包装。

这个方法主要的作用是：

1.当传入的View不为空时，如果我们在布局中发现了CoordinatorLayout布局，那么返回的View就是CoordinatorLayout；

2.如果没有CoordinatorLayout布局，我们就先找到一个id为android.R.id.content的FrameLayout（这个布局是最底层的根布局），将View设置为该FrameLayout；

3.其他情况下就使用View的Parent布局一直到这个View不为空。

5. 点击登录，判断学号和密码的完整性和正确性，给出反馈信息

- 首先获取登录按钮的控件，注册监听器，当按钮被点击时，获取输入学号和密码出的两个TextInputLayout控件，并用getEditText().getText().toString()获取当前填入的学号和密码两个字符串number和password。

```
Button login = (Button)findViewById(R.id.signin);
login.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        TextInputLayout numberTextLayout = (TextInputLayout) findViewById(R.id.sid);
        TextInputLayout passTextLayout = (TextInputLayout) findViewById(R.id.password);
        String number = numberTextLayout.getEditText().getText().toString();
        String password = passTextLayout.getEditText().getText().toString();
```

- 然后用TextUtils.isEmpty()函数判断学号number是否为空，如果为空，则开启学号所在的输入控件的显示提示信息numberTextLayout.setErrorEnabled(true)，然后用setError()函数显示提示信息。如果学号不为空，密码为空，则关闭学号的显示提示信息，开启密码所在的输入控件的显示提示信息，并显示提示信息。如果两者都不为空，则两个都要关闭。然后用equal()函数判断学号和密码是否都正确，根据情况设定不同的message，然后弹出snackbar。同样要设置snackbar的按钮被点击时弹出toast提示信息。

```
if(TextUtils.isEmpty(number)){
    numberTextLayout.setErrorEnabled(true);
    numberTextLayout.setError("学号不能为空");
}
else if(TextUtils.isEmpty(password)){
    //numberTextLayout.setError("");
    numberTextLayout.setErrorEnabled(false);
    passTextLayout.setErrorEnabled(true);
    passTextLayout.setError("密码不能为空");
}
else {
    //passTextLayout.setError("");
    //numberTextLayout.setError("");
    numberTextLayout.setErrorEnabled(false);
    passTextLayout.setErrorEnabled(false);
```

```
else {
    //passTextLayout.setError("");
    //numberTextLayout.setError("");
    numberTextLayout.setErrorEnabled(false);
    passTextLayout.setErrorEnabled(false);

    String message;
    if(number.equals("123456") && password.equals("6666"))
        message = "登陆成功";
    else
        message = "学号或密码错误";
    Snackbar.make(rootView,message,Snackbar.LENGTH_SHORT)
            .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();
                }
            })
            .setActionTextColor(getResources().getColor(R.color.colorPrimary))
            .setDuration(3000)
            .show();
```

6. 点击注册按钮，根据身份提示**“XX注册功能尚未启用”**

- 首先获取注册按钮这个控件，然后注册监听器。一旦按钮被点击，用getCheckedRadioButtonId()函数获取选择“学生”或“教职工”的RatioGroup此时被选中的button的id，当然，此处返回依然不是原始的id，而应该是id在整个xml布局中的序号(?)。同样地，用getId()函数的返回值与其进行对比，以此来判断被选中的是“学生”还是“教职工”，以设定不同的message。之后将message传入snackbar弹出不同的提示消息。此处用的radioGroup、student和teacher控件，都是在前面获取过并声明成final的，可以理解是一个常量。

```
Button register = (Button)findViewById(R.id.regist);
register.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String message = "";
        int checkedId = radioGroup.getCheckedRadioButtonId();
        if(checkedId == student.getId())
            message = "学生注册功能尚未启用";
        else if(checkedId == teacher.getId())
            message = "教职工注册功能尚未启用";

        Snackbar.make(rootView, message, Snackbar.LENGTH_SHORT)
                .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();
                    }
                })
                .setActionTextColor(getResources().getColor(R.color.colorPrimary))
                .setDuration(3000)
                .show();
    }
});
```

## 4. 实验思考及感想

这个实验其实花了很多蛮多时间，虽然很多东西理论课上有讲，但真正应用起来并不像听起来那么简单。很多地方都是细节问题，很繁琐，考细心程度，考耐性。而且很多地方其实都得靠自己利用搜索引擎去查找用法，去查找原因。但是做完很有成就感。

其中主要有两个点，一个是用方法获取控件的id通常得到的不是原始的id，不能直接将获取到的id转换为string后与原始id进行比较，这样不能得到正确的比较结果。还有一个是string的比较问题，一开始很直接地就用c++的==符号进行比较，但是java中不是这样用的。而需要用equal函数来比较。这是踩到的两个隐蔽的坑。

还有就是，整个实验一开始很多时间是处于懵逼和查资料的状态，后来理清头绪了，知道哪个地方该用什么，然后按照实验要求一步一步去实现。下次要吸取教训，不要多点同时用功，一个一个点逐一击破会更加有效率。