

中山大学移动信息工程学院本科生实验报告

课程名称：移动应用开发

任课教师：郑贵锋

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、实验内容

实现一个 Android 应用，要求如下：

1. 该界面为应用启动后看到的第一个界面 (如右图 1)
2. 输入学号和密码的控件要求用 TextInputLayout 实现
3. 点击图片，弹出对话框如图 2
 - 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”，如右图：



图



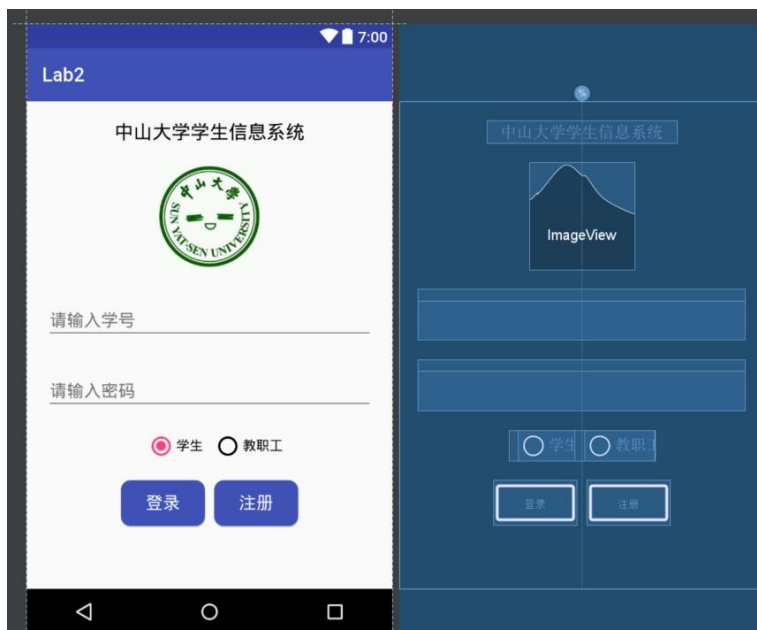
6. 点击注册按钮
如果切换选项时，RadioButton 选中的是“学生”，那么弹出 Snackbar 信息“学生注册功能尚未启用”，如果选中的是“教职工”，那么弹出 Snackbar 信息“教职工注册功能尚未启用”。如右图：



四、课堂实验结果

(一) 实验截图：

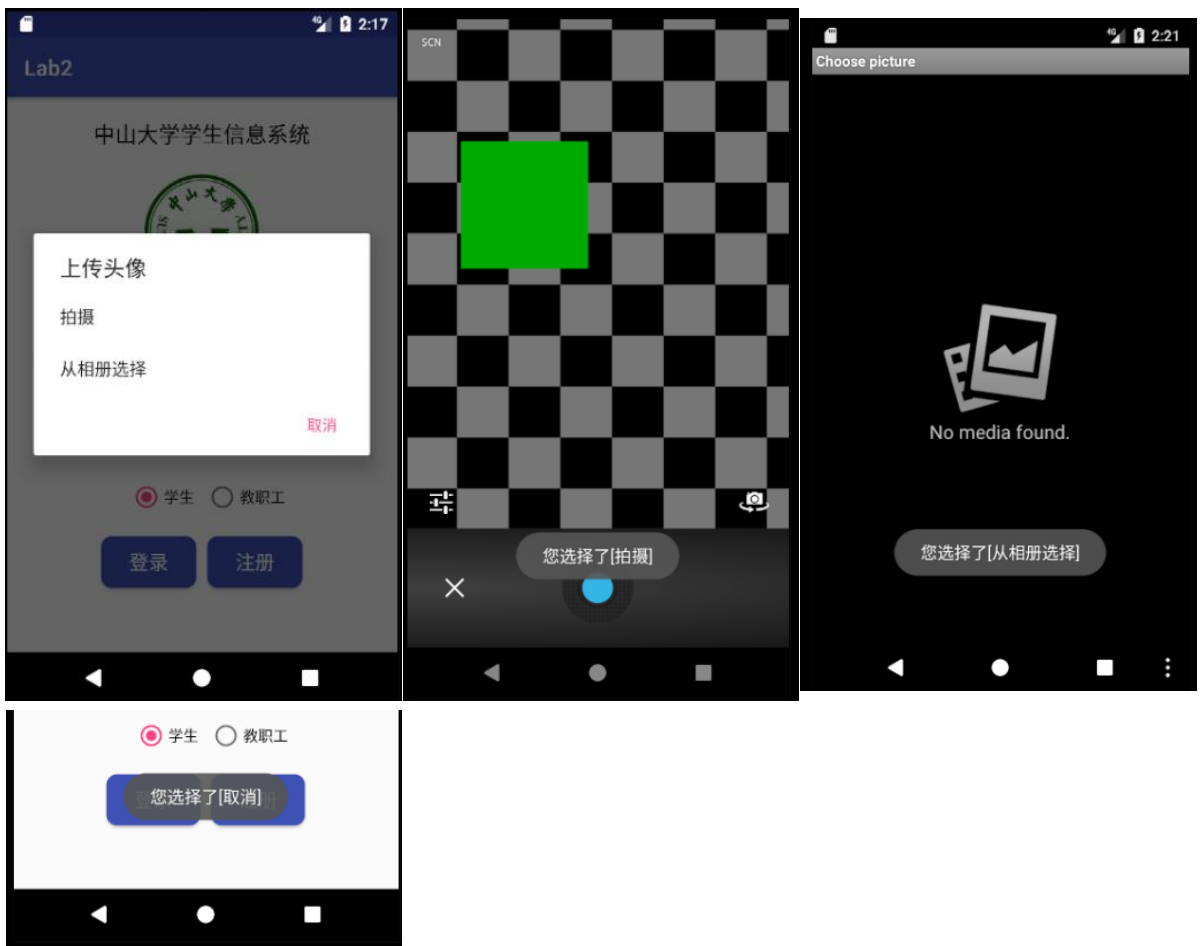
1. 界面效果如下：



2. 虚拟机上效果如下：



1) 点击图片，弹出对话框，此处的拍摄和从相册选择都是额外实现的功能。



2) 切换 RadioButton 的选项，弹出 Snackbar 提示“您选择了 xx”



3) 点击按钮效果



(二) 关键步骤：

1. 本次实验需要使用到 `TextInputLayout` 所以需要在修改先对应的 `build.gradle` 文件中的配置

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 26 //版本对应
    buildToolsVersion '26.0.0' //版本对应
    defaultConfig {
        applicationId "cn.chonor.lab2"
        minSdkVersion 19
        targetSdkVersion 26 //版本对应
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}

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-v7:26.0.0-alpha1' //版本对应
    compile 'com.android.support:design:26.0.0-alpha1' //版本对应
    compile 'com.android.support.constraint:constraint-layout:1.0.2' //TextInputLayout 控件
    testCompile 'junit:junit:4.12'
}

```

2. 删掉之前的 EditText 控件和对应的 TextView 控件，替换为 TextInputLayout 控件，代码修改部分如下(一个 TextInputLayout 控件):

```

<android.support.design.widget.TextInputLayout
    android:id="@+id/textInputLayout"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginEnd="20dp"
    android:layout_marginLeft="20dp"
    android:layout_marginRight="20dp"
    android:layout_marginStart="20dp"
    android:layout_marginTop="20dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/imageView"
    app:layout_constraintVertical_bias="0.0"
    >

    <EditText
        android:id="@+id/textid"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="@string/user_name_hit"
        android:inputType="number" />
</android.support.design.widget.TextInputLayout>

```

3. 界面修改完毕之后开始添加事件处理代码。

- i. ImageView 点击事件

- 首先先引入控件，此处我们需要引入 ImageView 控件

```
mImage = (ImageView) findViewById(R.id.imageView);
```

- 接着设置对于点击事件的监听

```

mImage.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        AlertDialog dialog=builder.create();//完成创建 AlertDialog 并显示
        dialog.show();
    }
});

```

- 此处因为点击图片之后我们需要弹出一个 AlertDialog 对话框，所需要新建一个 AlertDialog 并设置好其选择内容，AlertDialog 对话框创建如下

```
public void alertDialog_build() {
```

```
//alertdialog 初始化
builder.setTitle("上传头像");
final String[] Items={"拍摄","从相册选择"};
builder.setItems(Items,new DialogInterface.OnClickListener() {
    @Override //使用 setItems 构建选择列表 并增加点击检测
    public void onClick(DialogInterface dialogInterface, int i) {
        Toast.makeText(getApplicationContext(), "您选择了["+Items[i]+"",
Toast.LENGTH_SHORT).show();
    }
});
builder.setNegativeButton("取消", new DialogInterface.OnClickListener() {
    @Override //设置取消按钮动作
    public void onClick(DialogInterface dialogInterface, int i) {
        Toast.makeText(getApplicationContext(), "您选择了[取消]",
Toast.LENGTH_SHORT).show();
    }
});
builder.setCancelable(true); //允许取消
}
```

ii. RadioGroup 切换的提示

- 考虑到我们在 RadioGroup 所选择的部分在之后 button 点击事件中还要使用，所以设置一个全局的 String 变量用来记录 RadioGroup 点击内容

```
String rb_string="学生"; //初始化为默认选项学生
```

- 之后引入 RadioGroup 控件和 RadioButton 控件

```
mRB = (RadioGroup) findViewById(R.id.radioGroup);
rb_student = (RadioButton) findViewById(R.id.radioButton2);
rb_teacher = (RadioButton) findViewById(R.id.radioButton1);
```

- 然后设置 RadioGroup 改变选择的事件监听，此处弹出 Snackbar，选择 Snackbar 的取消时还需要弹出 Toast

```
mRB.setOnCheckedChangeListener(new RadioGroup.OnCheckedChangeListener() {
    @Override
    public void onCheckedChanged(RadioGroup group, int checkedId) {
        if (rb_student.getId() == checkedId) { //保存当前选择
            rb_string = rb_student.getText().toString();
        }
        if (rb_teacher.getId() == checkedId) {
            rb_string = rb_teacher.getText().toString();
        }
        Snackbar.make(button_init, "您选择了"+rb_string, Snackbar.LENGTH_SHORT)
            .setAction("确定", new View.OnClickListener() {
                @Override
                public void onClick(View view) {
                    Toast.makeText(getApplicationContext(), "Snackbar 的确定按钮
被点击了", Toast.LENGTH_SHORT).show();
                }
            })
            .setActionTextColor(getResources().getColor(R.color.colorPrimary))
            .show();
    }
});
```

```

    }
});

```

iii. Button 点击事件的监听

- 引入 button 控件

```

button_login = (Button) findViewById(R.id.button);
button_init = (Button) findViewById(R.id.button2);

```

- 登录按钮事件监听,此处需要使用的例如 TextInputLayout 等已经引入只需要根据相应情况弹出对应提示

```

button_login.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        if(et_id.getText().toString().length()==0){ //先判断 学号
            til_id.setError("学号不能为空");
            til_id.setErrorEnabled(true);
        }
        else if(et_password.getText().toString().length()==0) { //再判断 密码
            til_password.setError("密码不能为空");
            til_password.setErrorEnabled(true);
        }
        else if(et_id.getText().toString().equals("123456") &&
et_password.getText().toString().equals("6666")) { //如果都不为空 判断是否中输入正
确
            Snackbar.make(button_init, "登录成功", Snackbar.LENGTH_SHORT)
                .setAction("确定", new View.OnClickListener() {
                    @Override
                    public void onClick(View view) {
                        Toast.makeText(getApplicationContext(), "Snackbar 的确定按钮被点击了", Toast.LENGTH_SHORT).show();
                    }
                })
                .setActionTextColor(getResources().getColor(R.color.colorPrimary))
                .show();
        }
        else if(et_id.getText().toString().length()!=0 &&
et_password.getText().toString().length()!=0) { //输入不正确
            Snackbar.make(button_init, "学号或密码错误", Snackbar.LENGTH_SHORT)
                .setAction("确定", new View.OnClickListener() {
                    @Override
                    public void onClick(View view) {
                        Toast.makeText(getApplicationContext(), "Snackbar 的确定按钮被点击了", Toast.LENGTH_SHORT).show();
                    }
                })
                .setActionTextColor(getResources().getColor(R.color.colorPrimary))
                .show();
        }
    }
}

```

```
}  
});
```

(三) 实验中遇到的困难和解决思路:

1. 最开始写 java 文件的时候发现想用很多控件名字打出来都是 cannot resolve symbol, 然后百度半天 import, 然后发现可以直接使用 alt+enter 自动补全
2. 一开始修改好界面之后, 在虚拟机中运行发现只要一开启 app, 屏幕焦点就自动集中在密码输入框上, 然后输入密码的提示消失。于是百度之, 最终解决方法是在 xml 文件中使得一开始把焦点移走, 此处选择的是把焦点移到完全没有影响的地方上, 在 xml 文件中增加如下两句:

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

增加之后显示效果正常。

五、课后实验结果

1. 实验中要求点击照片弹出拍照和从相册选择这两个选项, 然而并不要求实现, 只要求输出提示, 于是百度了一下如何调出相机和相册, 此处只需要在 AlertDialog 的 Items, 点击中多增加几句即可, 使用 Intent 打开相机或者相册, 然后传入参数调用处理函数。

```
if(i==0){//额外 打开相机  
    Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);  
    startActivityForResult(intent, CAMERA);  
}else{//额外 打开相册  
    Intent intent = new Intent(Intent.ACTION_PICK);  
    intent.setType("image/*");  
    startActivityForResult(intent, IMAGE);  
}
```

处理函数针对相机和相册分别处理, 此处的相机处理比较简单只需要一句话丢给 ImageView 显示就可以了, 此处因为我们不储存所以很简单, 从相册获取, 这个代码是像百度学习的, 因为不同的系统需要不用的调用处理函数, 所以此处也是不太清楚, 于是就学习了适合 API 版本 26 的做法, 使用输入流处理。

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
    if (requestCode == CAMERA && resultCode == RESULT_OK) {  
        Bitmap photo = (Bitmap) data.getExtras().get("data");  
        mImage.setImageBitmap(photo); //设置图片  
        para = mImage.getLayoutParams(); // 设置自动宽高  
        para.height = height;  
        para.width = width;  
        mImage.setLayoutParams(para);  
  
    }else if(requestCode == IMAGE && resultCode == RESULT_OK){  
        try { //此处提示需要抛出异常 所以加了  
            InputStream inputStream = getContentResolver().openInputStream(data.getData());  
            Bitmap photo = BitmapFactory.decodeStream(inputStream); //使用输入流转化图片  
            mImage.setImageBitmap(photo);  
            para = mImage.getLayoutParams(); // 设置自动宽高
```



```

        para.height = height;
        para.width = width;
        mImage.setLayoutParams(para);
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }
}

super.onActivityResult(requestCode, resultCode, data);
}

```

此处需要设置固定宽高，所以要先获取图片宽高，测试了百度上的两种方式最后发现用 View.MeasureSpec 测量宽高，之后设置图片的时候 LayoutParams 固定照片大小比较方便，一下为测量图片宽高代码

```

int i = View.MeasureSpec.makeMeasureSpec(0, 0);
int j = View.MeasureSpec.makeMeasureSpec(0, 0);
mImage.measure(i, j);
height=mImage.getMeasuredHeight();
width=mImage.getMeasuredWidth();

```

2. TextInputLayout 控件在输入结束时直接判断是不是输了空账号密码，而不是需要经过 button

- TextInputLayout 和 EditText 控件

```

et_id = (EditText) findViewById(R.id.textid);
et_password = (EditText) findViewById(R.id.textpassword);
til_id = (TextInputLayout) findViewById(R.id.textInputLayout);
til_password = (TextInputLayout) findViewById(R.id.textInputLayoutI);

```

- 增加输入框输入的事件监听(此处展示输入学号密码同理)，此处同时使得提示更加友好，选择教职工的时候提示从学号 xxxx 变成教职工号 xxxx

```

et_id.addTextChangedListener(new TextWatcher() {
    @Override
    public void beforeTextChanged(CharSequence charSequence, int i, int i1, int i2) {
    }
    @Override
    public void onTextChanged(CharSequence charSequence, int i, int i1, int i2) {
    }
    @Override
    public void afterTextChanged(Editable editable) { //额外//输入完毕自主判断
        if (editable.length() == 0) {
            til_id.setErrorEnabled(true);
            til_id.setError(rb_string_hit+"不能为空");
        } else {
            til_id.setErrorEnabled(false);
        }
    }
});

```

- 此次同时调整了一下提示，使得提示更友好：

请输入教职工号	请输入教职工号 123456
请输入密码	请输入密码
<input type="radio"/> 学生 <input checked="" type="radio"/> 教职工	<input type="radio"/> 学生 <input checked="" type="radio"/> 教职工
<input type="button" value="登录"/> <input type="button" value="注册"/>	<input type="button" value="登录"/> <input type="button" value="注册"/>
您选择了教职工	您选择了教职工

切换的时候会更改提示。

- 既然做了登录，所以做了一个登录后转跳的界面。页面转跳也很简单通过 Intent 就可以实现转跳，不过我们需要写一个新的 class 和一个新的 xml 布局文件，作为转跳用，此处增加的为学生登录成功后才进行转跳，同时使用 Timer 和 TimerTask 进行转跳延时(只是为了留存提示)。

- 此处需要先修改两个地方

首先在 AndroidManifest 中注册我们这个心的 class

```
<activity android:name=".AfterLogin"></activity>
```

然后在 class 中和 xml 文件中分别设定绑定

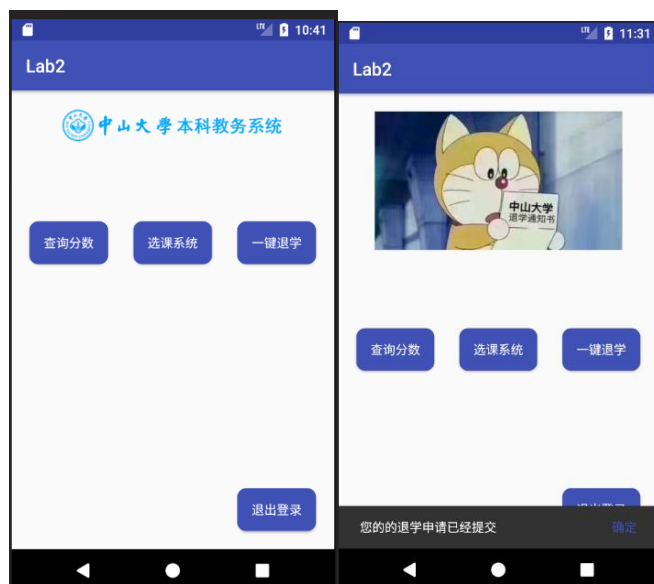
```
setContentView(R.layout.after_login);
```

```
tools:context="cn.chonor.lab2.AfterLogin"
```

- 新增的转跳以及延时代码学生登录成功后才进行转跳

```
if(rb_string.equals("学生")){
    Timer timer = new Timer();
    timer.schedule(new TimerTask(){
        @Override
        public void run() {
            Intent i = new Intent(MainActivity.this, AfterLogin.class);
            startActivity(i);
        }
    }, 2000); //延时 2s 执行
```

- 转跳后界面如下(反正除了退学按钮都不行 😊) :



- 转跳界面后设置退出，之后继续转跳回主登录界面，此处代码没有多少变化，所以在展示。

六、实验思考及感想

这次实验主要是写事件监听的代码，界面布局基本没有变动，从写的时候的感觉来说，这个事件处理还是很简单的，本质上就是一个注册监听，一旦监听到这个动作之后就会调用相应的函数，我们只要写相应的执行部分就行了，注册监听的方式也是比较相似，就是动不动就要个 `import` 真的麻烦。实验中还涉及了两种消息提示模式，这两种的参数也比较固定使用起来也比较简单。还是自己做点拓展好玩，那个打开相机比较简单，只需要用 `Intent` 即可打开相机返回也可以直接使用 `Bitmap` 来实现显示，但是打开相册返回图片就比较麻烦了，还需要设计到转换，一开始套用相机的直接出运行 `bug`，之后百度了才得到了比较好用的方式。拓展的时候还涉及了一下新的页面转跳，觉得这个转跳是挺简单，但是用来写 APP 的时候也是很重要的。