

中山大学移动信息工程学院本科生实验报告

(2016 年秋季学期)

课程名称：移动应用开发

任课教师：郑贵锋

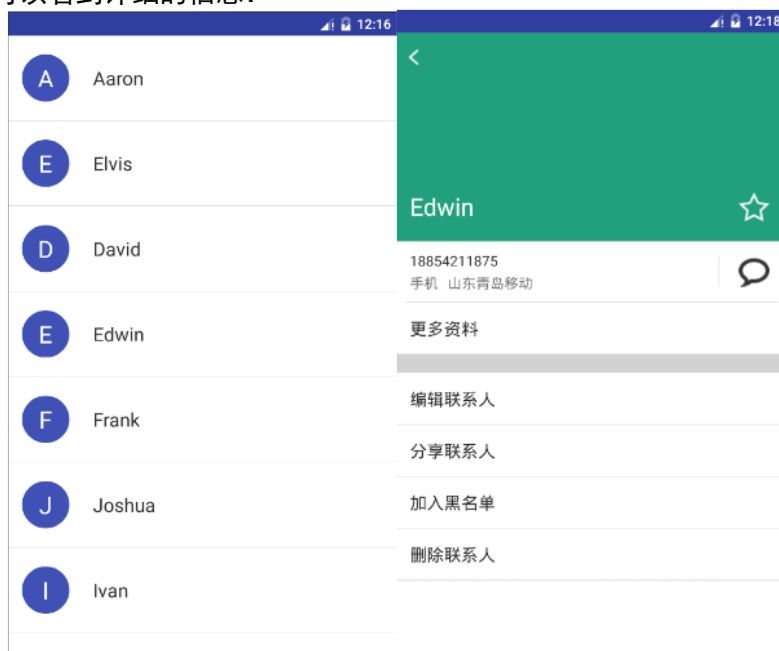
年级	14	专业 (方向)	物联网
学号	14353259	姓名	商家煜
电话	13211143626	Email	shangjy3@mail2.sysu.edu.cn
开始日期	2016.9.22	完成日期	2016.9.22

一、 实验题目

Intent、Bundle 的使用和 ListView 的应用

二、 实现内容

本次实验模拟实现一个通讯录，有两个界面，第一个界面用于呈现通讯录，如下所示：
点击任意一项后，可以看到详细的信息：



实验要求：

布局方面的要求：

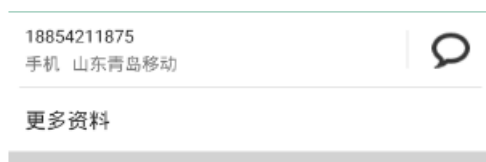
1、通讯录界面

每一项为一个圆圈和一个名字，圆圈与名字均竖直居中。圆圈中为名字的首字母，首字母要处于圆圈的中心，首字母为白色，名字为黑色，圆圈的颜色自定义即可，建议用深色的颜色，否则白色的首字母可能看不清。

2、联系人详情界面顶部

顶部的背景色在通讯录数据中已经给出，每个人对应有一种特定的颜色，这块背景色占整个界面的1/3，返回图标处于这块 View 的左上角，联系人名字处于左下角，星标处于右下角，它们与边距都有一定距离，自己调出合适的距离即可。需要注意的是，返回图标与名字左对齐，名字与星标底边对齐。这一块建议大家去看一下 RelativeLayout 的使用。

3、联系人详情界面中部



使用的黑色 argb 编码值为#D5000000，稍微偏灰色一点的“手机”、“山东青岛移动”的 argb 编码值为#8A000000。注意，电话号码那一栏的下边有一条分割线，argb 编码值为#1E000000，右边聊天符号的左边也有一条分割线，argb 编码值也是#1E000000，这条分割线要求高度与聊天符号的高度一致，并且竖直居中。字体的大小看着调就可以了。“更多资料”底部的分割线高度自己定，argb 编码值与前面的分割线一致。

4、联系人详情页面底部



5、两个界面顶部都没有标题栏，要用某些方法把它们去掉。

逻辑方面的要求：

1、点击通讯录中的某一个联系人会跳转到联系人详情界面，呈现该联系人的详细信息；长按通讯录中的联系人会弹出对话框询问是否删除该联系人，点击确定则删除该联系人，点击取消则对话框消失。

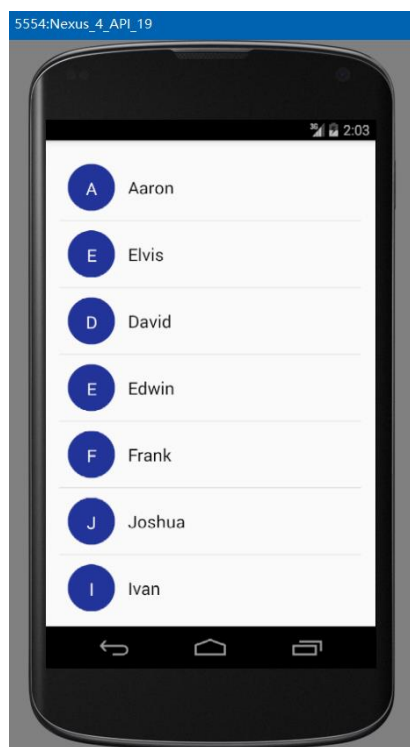


注意对话框中的人名为被长按的联系人。

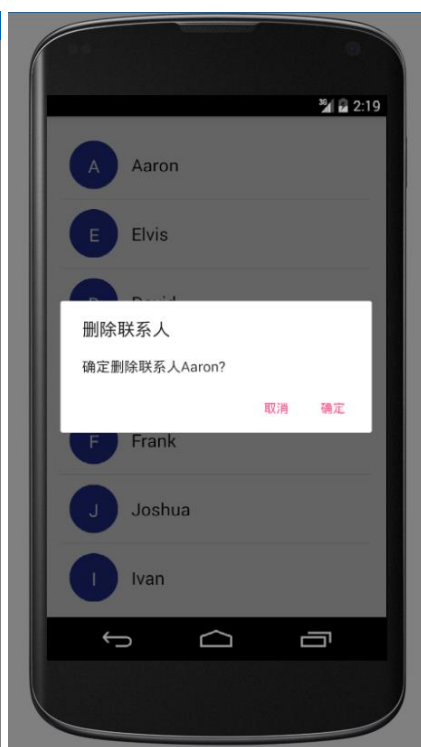
2、联系人详情界面中点击返回图标会返回上一层，点击星标会切换状态，如果原先是空心星星，则会变成实心星星；如果原先是实心星星，则会变成空心星星。

三、 课堂实验结果

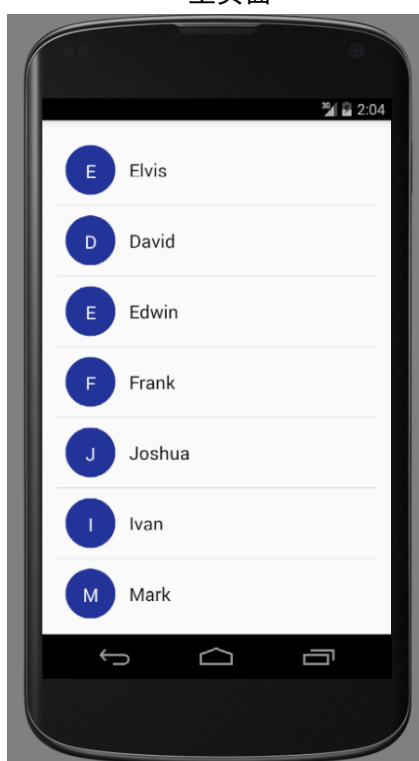
(1) 实验截图



主页面



长按删除



删除后



详细信息

(2) 实验步骤以及关键代码

1) Activity 设置

```
<activity android:name=".MainActivity">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>

<activity android:name=".SecondActivity">
    <intent-filter>
        <action android:name="com.litreily.SecondActivity" />

        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

图：两个 Activity

创建两个 Activity 分别来表示主界面和详情界面，将主界面设置为 LAUNCHER 状态表示开启程序后显示，将详情界面设置为 DEFAULT 状态等待调用。

2) 主界面设置

主页面的 XML 中只在 LinearLayout 下添加了一个 ListView，因此不再截图，然后对该 ListView 进行 java 代码编写。

```
final List<Map<String, Object>> data = new ArrayList<>();
String[] num = {"A", "E", "D", "E", "F", "J", "I",
               "H", "J", "P"};
final String[] name = {"Aaron", "Elvis", "David", "Edwin", "Frank",
                      "Joshua", "Ivan", "Mark", "Joseph", "Phoebe"};
for(int i = 0; i < 10; i++) {
    Map<String, Object> temp = new LinkedHashMap<>();
    temp.put("num", num[i]);
    temp.put("name", name[i]);
    data.add(temp);
}
```

图:ListView

首先创建一个名为 data 的 ArrayList 类型来存储一个 map 类型（这里表示为两个相关的字符串）。创建两个已经确定的字符串，通过一个 for 循环，将两个字符串通过“key 值”（即 num 和 name）来向一个临时的 map 类型插入，然后调用 add 来插入到 data 中。

```
final ListView listview = (ListView) findViewById(R.id.contacts_list);
final SimpleAdapter simpleAdapter = new SimpleAdapter(this, data, R.layout.item,
    new String[]{"num", "name"}, new int[]{R.id.num, R.id.name});
listview.setAdapter(simpleAdapter);
```

图：创建 listview 实例

其次，通过函数找到制定的 ListView 并绑定，然后创建 SimpleAdapter 类型，将刚创建好的 data 传入，并新建一个 layout，来设定 data 的数据应该如何排布（这里就简单的通过 LinearLayout 的水平排布将 num 和 name 设置成要求的样式）。第四个参数是一个 String[] 类型的参数，该参数决定提取 Map<String, ?>对象中哪些 key 对应的 value 来生成列表项，第 5 个参数是一个 int[] 类型的参数，该参数决定填充哪些组件。

```
listview.setOnItemClickListener((parent, view, position, id) -> {
    Intent intent = new Intent(MainActivity.this, SecondActivity.class);
    Bundle bundle = new Bundle();
    bundle.putInt("data", position);
    intent.putExtras(bundle);
    startActivity(intent);
});
```

图：页面跳转和传递参数

然后通过调用 ListView 的方法，来设置监听，当被短暂按下时，创建一个 intent 类型，来进行显示页面跳转，其中第一个参数表示为当前 Activity，第二个参数为需要跳转到的 Activity。之后再创建一个 Bundle 实例来对需要传送的参数进行封装。其中传递的信息为当前按下的位置（从 0 开始），并将“暗号”设置为“data”，随后通过 putExtras 方法加入到 intent 实例中，然后 startActivity。

```
final AlertDialog.Builder alertDialog = new AlertDialog.Builder(this);
listview.setOnItemLongClickListener((parent, view, position, id) -> {
    AlertDialog alertDialog = alertDialog.setTitle("删除联系人").setMessage("确定删除联系人" + data.get(position).get("name") + "?"
    ).setPositiveButton("确定",
        (dialog, which) -> {
            Toast.makeText(getApplicationContext(), "对话框“确定”按钮被点击", Toast.LENGTH_SHORT).show();
            data.remove(position);
            simpleAdapter.notifyDataSetChanged();
        }).setNegativeButton("取消",
        (dialog, which) -> {
            Toast.makeText(getApplicationContext(), "对话框“取消”按钮被点击", Toast.LENGTH_SHORT).show();
        }).create();
    alertDialog.show();
});
```

图：长按弹出对话框

长按的设置中，创建一个简单对话框，并将其弹出，如果用户选择确定按钮，则调用方法 remove 将长按的位置移出，然后调用方法 notifyDataSetChanged() 对当前 simpleAdapter 进行刷新。

3) 详情界面设置

● XML 界面

```
android:layout_width="match_parent"
android:layout_height="0dp"
android:layout_weight="1">
```

图：设置比例

由于详情页面需要上面的标签占据全界面的 1/3，所以在 XML 界面中使用 layout_weight 来设置即可。

```
<ImageView
    android:layout_width="fill_parent"
    android:layout_height="2dp"
    android:background="#1E000000"
    android:id="@+id/line1"
    android:layout_below="@+id/where"></ImageView>
```

图：设置分割线

分割线通过简单的 ImageView 技巧设置，通过设定 ImageView 中的 layout_height 来设置分割线的粗细，以及设置背景色来满足题目要求。布局采用 RelativeLayout 来进行布局。通过设置每个元件相对于父亲控件以及相对于其他同级控件的位置标签来达到题目要求。

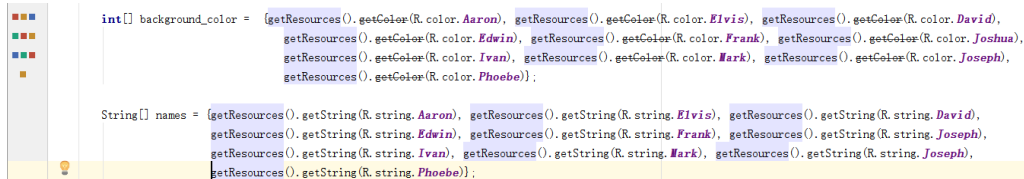
其他地方没有什么特殊之处，在此就不再赘叙。

- java 代码

```
Intent intent = this.getIntent();
Bundle bundle = intent.getExtras();
int data = bundle.getInt("data");
```

图：接受参数

首先在新的 activity 中获取传递过来的参数，通过之前设置的密码作为 getIntent 的参数即可得到参数。



```
int[] background_color = {getResources().getColor(R.color.Aaron), getResources().getColor(R.color.Elvis), getResources().getColor(R.color.David),
getResources().getColor(R.color.Edwin), getResources().getColor(R.color.Frank), getResources().getColor(R.color.Joshua),
getResources().getColor(R.color.Ivan), getResources().getColor(R.color.Mark), getResources().getColor(R.color.Joseph),
getResources().getColor(R.color.Phoebe)};

String[] names = {getResources().getString(R.string.Aaron), getResources().getString(R.string.Elvis), getResources().getString(R.string.David),
getResources().getString(R.string.Edwin), getResources().getString(R.string.Frank), getResources().getString(R.string.Joseph),
getResources().getString(R.string.Ivan), getResources().getString(R.string.Mark), getResources().getString(R.string.Joseph),
getResources().getString(R.string.Phoebe)};
```

图：设置数组来进行选择

首先在 string 和 color 中设置好固定的需要的值，然后设置数组以方便调用和以后的数据更新。

```
RelativeLayout label = (RelativeLayout) findViewById(R.id.label);
TextView text = (TextView) findViewById(R.id.name);
TextView phone_number = (TextView) findViewById(R.id.phone_number);
TextView phone = (TextView) findViewById(R.id.phone);
TextView where = (TextView) findViewById(R.id.where);
```

```
label.setBackgroundColor(background_color[data]);
text.setText(names[data]);
phone_number.setText(number[data]);
phone.setText(type[data]);
where.setText(address[data]);
```

图：通过得到的参数设置页面内容

然后通过得到的参数来得到对应数组中的值，最后再通过绑定的模块进行函数的调用设置页面中需要的值。

```
final Button star = (Button) findViewById(R.id.star);
star.setOnClickListener(new View.OnClickListener() {
    int flag = 1;
    @Override
    public void onClick(View v) {
        flag = ~flag;
        if(flag == 1)
            star.setBackgroundResource(R.mipmap.full_star);
        else
            star.setBackgroundResource(R.mipmap.empty_star);
    }
});
```

图：按钮转跳会主界面

判断星星的按钮是否被按下以改变当前的星星图标，通过设置一个 flag 来进行改变星星是 full 还是 empty。

```

ListView listView = (ListView) findViewById(R.id.contacts_list);
String[] name = {"编辑联系人", "分享联系人", "加入黑名单", "删除联系人"};
listView.setAdapter(new ArrayAdapter<String>(this, android.R.layout.simple_list_item_1, name));

Button button = (Button) findViewById(R.id.back);
button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        finish();
    }
});

```

图：ListView

基本同样的方式来设置一个 listView 以显示 listview 所需要的信息,然后同样当返回按钮被按下的时候调用 finish 函数来结束这个界面

(3) 实验遇到困难以及解决思路

实验过程中对于对话框中获取联系人姓名的显示当时卡了比较久，因为对 map 的方法不是很了解，随后通过查阅相关的资料，知道了通过 get（）方法可以得到其中指定位置指定 key 值的 value，然后就很好的写出了对话框。

四、 实验思考及感想

实验中考虑到联系人有可能需要根据不同的情况进行更改，因此采用了数组的方式来进行对已有的数据库中的值进行调用的方法来存储，再通过传递来的参数，从数组中或许需要的值填充到界面中，这样子方便了以后软件的控制和维护。

实验过后将继续了解上课所教到的其他控件的使用方式，为下一次的实验课做准备