中山大學
SUN YAT-SEN UNIVERSITY

《手机平台应用开发》

Intent、Bundle 的使用和 ListView 的应用

学 院 名 称 ： 数据科学与计算机学院

成　　员 ： 陈伟宸

时　　间 ： 2016 年 10 月 3 日

一、实验环境

操作系统：Windows 10

IDE：Android Studio 2.1.2

二、实验过程

1. 新建一个 empty activity

2. 在 activity_main.xml 中添加一个 ListView 用于显示名字的首字母与名字

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="16dp"
    android:paddingTop="16dp"
    tools:context="com.example.administrator.contacts.MainActivity">

    <ListView
        android:id="@+id/contacts_list"
        android:layout_width="match_parent"
        android:layout_height="match_parent">
    </ListView>
</RelativeLayout>
```

3. 新建 Contact.java，在其中创建 Contact 类，用于保存联系人的姓名，号码，归属地等信息。由于之后要将该类的信息在 activity 之间传递，因此实现了 Parcelable 接口

4.

```java
public class Contact implements Parcelable{
    private String name;
    private String phone_number;
    private String attribution;
    private String background_color;

    public Contact() {}

    public Contact(String name, String phone_number, String attribution, String background_color) {
        this.name = name;
        this.phone_number = phone_number;
        this.attribution = attribution;
        this.background_color = background_color;
    }


    public String getName() { return  name; }

    public String getPhone_number() { return phone_number; }

    public String getAttribution() { return attribution; }

    public String getBackground_color() { return background_color; }

    @Override
    public int describeContents() { return 0; }

    @Override
    public void writeToParcel(Parcel dest, int flags) {...}

    public static final Parcelable.Creator<Contact> CREATOR = new Creator<Contact>() {...};
}
```

5. 自定义 Adapter。新建 ContactsAdapter.java，在其中创建 ContactAdapter 类，继承 BaseAdapter 类，并实现其方法

```
public class ContactsAdapter extends BaseAdapter {

    private List<Contact> contacts;
    private Context context;

    public ContactsAdapter(Context context, List<Contact> contacts) {...}

    @Override
    public int getCount() {...}

    @Override
    public Object getItem(int position) {...}

    @Override
    public long getItemId(int position) { return position; }

    @Override
    public View getView(int position, View view, ViewGroup parent) {...}

    private class ViewHolder {...}
}
```

6. 新建 circle.xml 和 contact.xml，设计通讯录中联系人的样式。其中，在 circle.xml 中定义包含联系人姓名首字母的背景圆圈，使用的方法是将 shape 声明为 oval，并将它的 width 和 height 设置为相等的值即可得到圆

```xml
<?xml version="1.0" encoding="UTF-8"?>
<shape
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="oval"
    android:useLevel="false" >
    <padding
        android:left="2dp"
        android:top="1dp"
        android:right="2dp"
        android:bottom="1dp" />
    <solid
        android:color="@color/colorPrimaryDark" />
    <!--<stroke-->
        <!--android:width="1dp"-->
        <!--android:color="@android:color/white" />-->
    <size android:width="60dp"
        android:height="60dp" />
</shape>
```

在 contact.xml 中做好排版

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal" android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:id="@+id/first_letter"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="@drawable/circle"
        android:textColor="#FFFFFF"
        android:gravity="center"
        android:textSize="30sp"
        android:text="A"
        android:layout_marginTop="20dp"
        android:layout_marginLeft="20dp"
        android:layout_marginBottom="20dp"/>

    <TextView
        android:id="@+id/name"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:textSize="28sp"
        android:text="Test"
        android:layout_marginTop="20dp"
        android:layout_marginLeft="20dp"/>
</LinearLayout>
```

7. 在 MainActivity.java 中导入联系人的信息，使用自定义的 ContactAdapter 设置 ListView

```java
String[] names = new String[] { "Aaron", "Elvis", "David", "Edwin", "Frank", "Joshua", "Ivan", "Mark", "Jo
String[] phone_numbers = new String[] { "17715523654", "18825653224", "15052116654", "18854211875", "13955
String[] attributions = new String[] {"江苏苏州电信", "广东揭阳移动", "江苏无锡移动", "山东青岛移动", "安徽合
String[] background_colors = new String[] {"BB4C3B", "c48d30", "4469b0", "20A17B"};

final List<Contact> contactList = new ArrayList<>();

for (int i = 0; i < 10; i++) {
    contactList.add(new Contact(names[i], phone_numbers[i], attributions[i], background_colors[i % 4]));
}

ListView listView = (ListView) findViewById(R.id.contacts_list);
final ContactsAdapter contactsAdapter = new ContactsAdapter(getApplicationContext(), contactList);
listView.setAdapter(contactsAdapter);
```

8. 在 MainActivity.java 中添加对 ListView 的 Item 的长按监听函数，实现删除功能

```
listView.setOnItemLongClickListener((parent, view, position, id) -> {
        deleteAlert.setMessage("确认删除联系人" + contactList.get(position).getName() + "?").setPositiveButton("确认", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                contactList.remove(position);
                contactsAdapter.notifyDataSetChanged();
            }
        }).setNegativeButton("取消", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
            }
        }).create().show();
        return true;
});
```

9. 新建 details.xml。在 details.xml 中设计联系人详情的页面。使用 layout_weight 属性设置比例

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="2"
    android:orientation="vertical">

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

用 RelativeLayout 将 message 图标的分割线长度设置成与 message 的高度相等

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="2">
    <LinearLayout
        android:id="@+id/phone_attribution"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="10dp">
        <TextView
            android:id="@+id/phone_number"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textColor="@color/colorDetailDark" />
        <TextView
            android:id="@+id/attribution"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textColor="@color/colorDetailLight" />
    </LinearLayout>
```

在操作部分使用 ListView 展示操作

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="13"
    android:orientation="vertical"
    android:layout_marginTop="15dp">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="@color/colorDetailDark"
        android:text="更多资料"
        android:layout_marginLeft="20dp" />
    <ImageView
        android:layout_width="match_parent"
        android:layout_height="20dip"
        android:background="@color/colorDetailDivider"
        android:layout_marginTop="15dp"/>
    <ListView
        android:id="@+id/operations"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</LinearLayout>
```

10. 新建 operations.xml，设计操作的样式

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <TextView
        android:id="@+id/operation"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="15dp"
        android:layout_marginBottom="15dp"
        android:textColor="@color/colorDetailDark" />

</LinearLayout>
```

11. 新建 DetailActivity.java，导入可选择的操作，并使用 SimpleAdapter 使它们在 ListView 中显示出来

```
String[] operations = new String[] {"编辑联系人", "分享联系人", "加入黑名单", "删除联系人"};
List<Map<String, Object>> data = new ArrayList<>();
for (int i = 0; i < operations.length; i ++) {
    Map<String, Object> temp = new LinkedHashMap<>();
    temp.put("operation", operations[i]);
    data.add(temp);
}
SimpleAdapter simpleAdapter = new SimpleAdapter(this, data, R.layout.operations, new String[] {"operation"}, new int[] {R.id.operation});
operationList.setAdapter(simpleAdapter);
```

12. 在 MainActivity.java 中添加对 ListView 的 Item 的点击监听函数，使其跳转到 DetailActivity 中，并传递被点击的 Item 的信息

```
listView.setOnItemClickListener((parent, view, position, id) -> {
        Intent intent = new Intent(MainActivity.this, DetailActivity.class);
        Bundle bundle = new Bundle();
        bundle.putParcelable("contact", contactList.get(position));
        intent.putExtras(bundle);
        startActivity(intent);
});
```

13. 在 DetailActivity.java 中实现获取到上一步传递过来的信息，并利用它们设置 details.xml 中的字段

```
Contact contact = (Contact)getIntent().getParcelableExtra("contact");

TextView name = (TextView)findViewById(R.id.name);
TextView phone_number = (TextView)findViewById(R.id.phone_number);
TextView attribution = (TextView)findViewById(R.id.attribution);
TextView background = (TextView)findViewById(R.id.background);
ImageView goback = (ImageView)findViewById(R.id.goback);
ImageView message = (ImageView)findViewById(R.id.message);
ImageView message_divider = (ImageView)findViewById(R.id.message_divider);
ListView operationList = (ListView)findViewById(R.id.operations);
final ImageView star = (ImageView)findViewById(R.id.star);

assert goback != null;
assert name != null;
assert phone_number != null;
assert attribution != null;
assert background != null;
assert message_divider != null;
assert message != null;
assert star != null;
```

```
name.setText(contact.getName());
phone_number.setText(contact.getPhone_number());
attribution.setText("手机 " + contact.getAttribution());
background.setBackgroundColor((Color.parseColor('#' + contact.getBackground_color())));
```

14. 在 DetailActivity.java 中实现返回功能

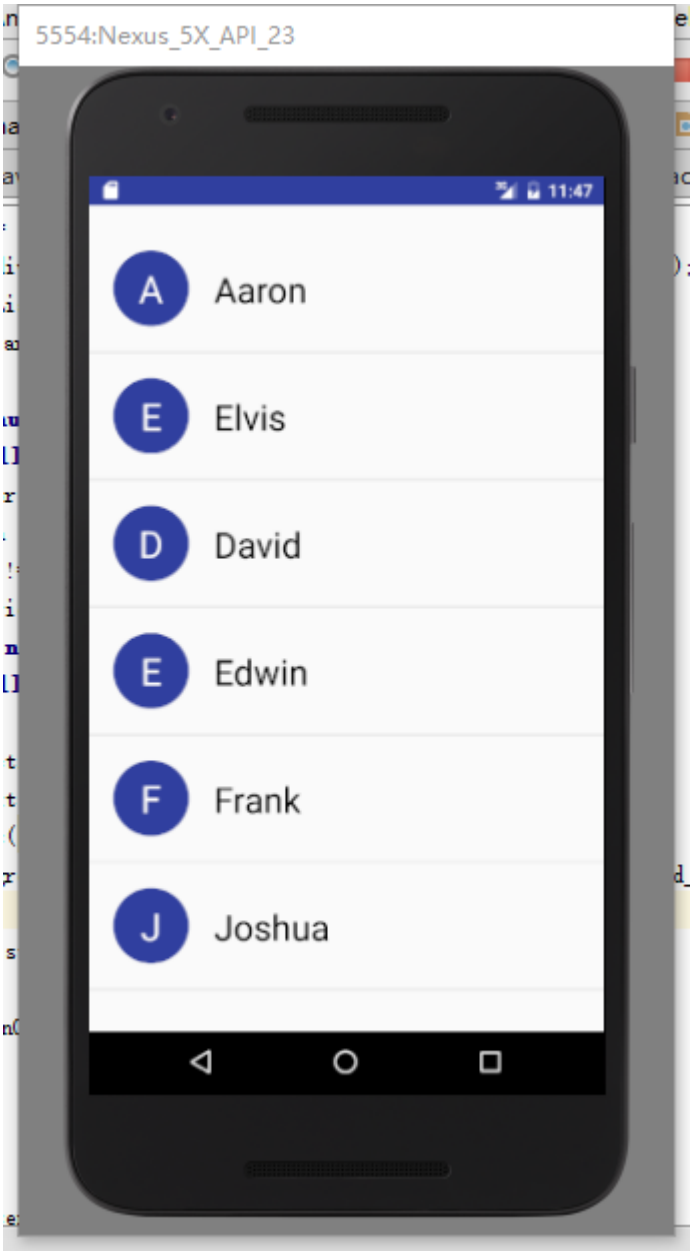
```
goback.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        finish();
    }
});
```

15. 在 DetailActivity.java 中实现点击 details.xml 中的星星使其变化的功能

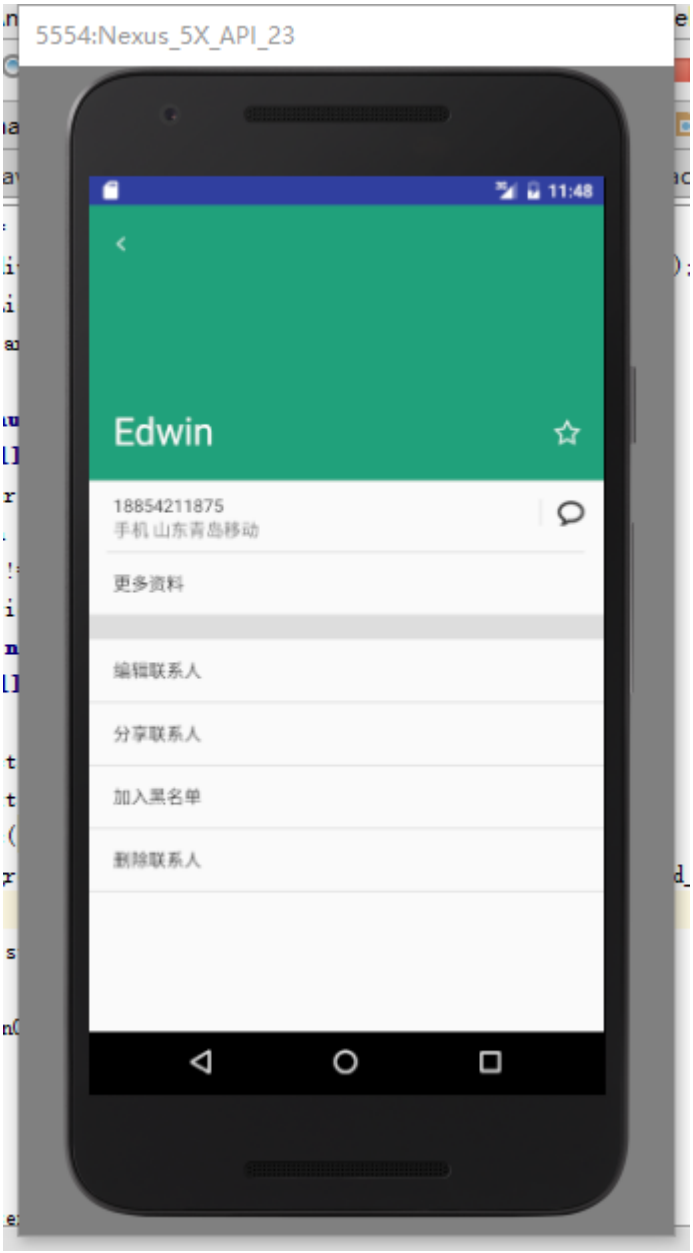
```
star.setOnClickListener((v) -> {
        if (star.getTag().toString().equals("0")) {
            star.setBackground(getDrawable(R.mipmap.full_star));
            star.setTag(1);
        }
        else {
            star.setBackground(getDrawable(R.mipmap.empty_star));
            star.setTag(0);
        }
});
```

三、实验结果

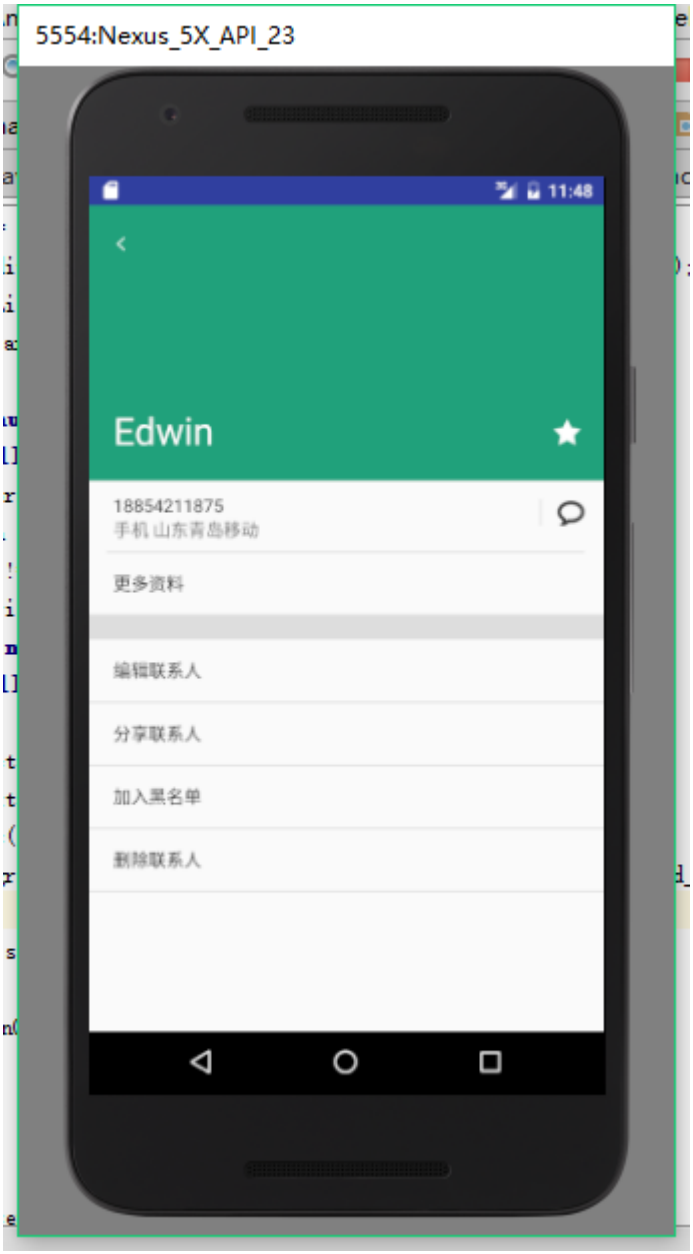
1. 运行后的界面


5554:Nexus_5X_API_23
11:47
A
Aaron
E
Elvis
D
David
E
Edwin
F
Frank
J
Joshua

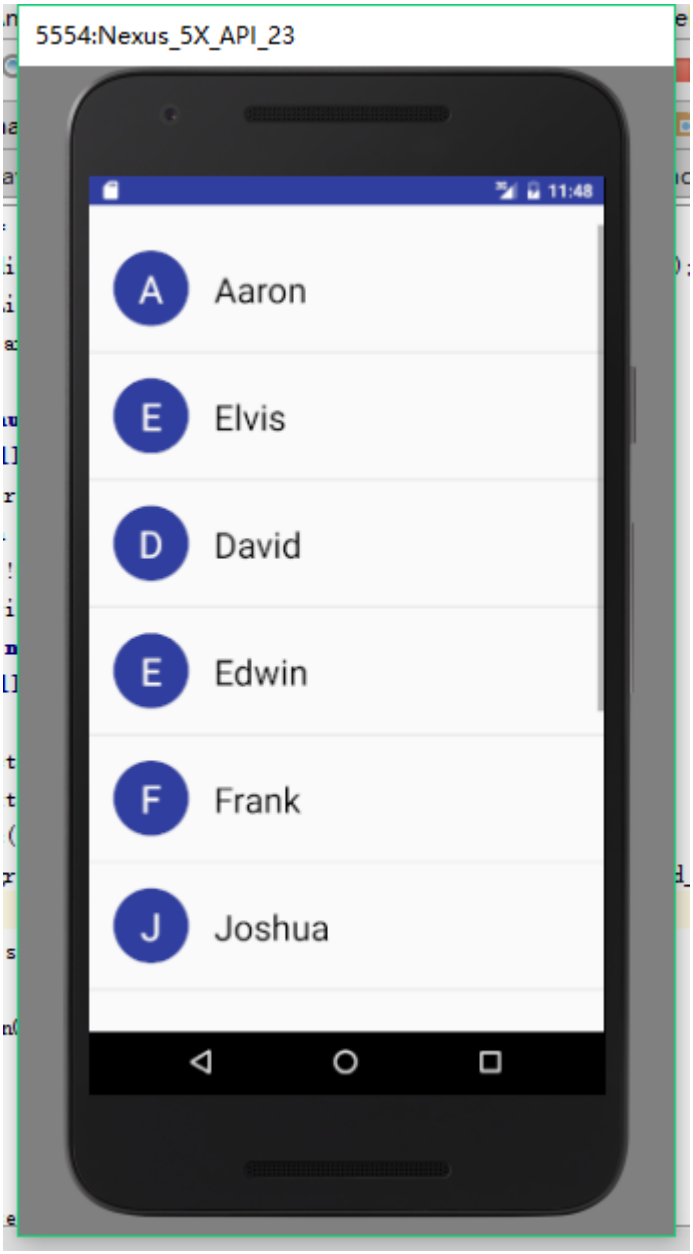
2. 点击任意一个联系人


5554:Nexus_5X_API_23
11:48
Edwin
18854211875
手机 山东青岛移动
更多资料
编辑联系人
分享联系人
加入黑名单
删除联系人

3. 点击星星


5554:Nexus_5X_API_23
11:48
Edwin
18854211875
手机 山东青岛移动
更多资料
编辑联系人
分享联系人
加入黑名单
删除联系人

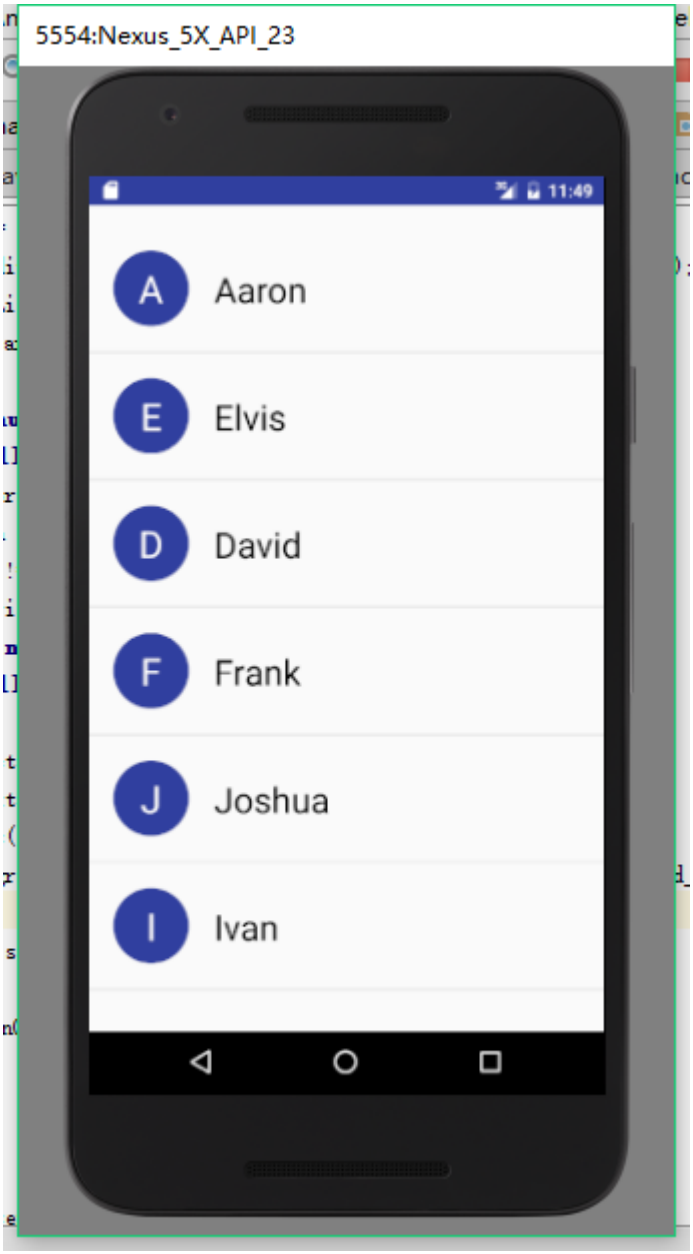
4. 点击返回按钮



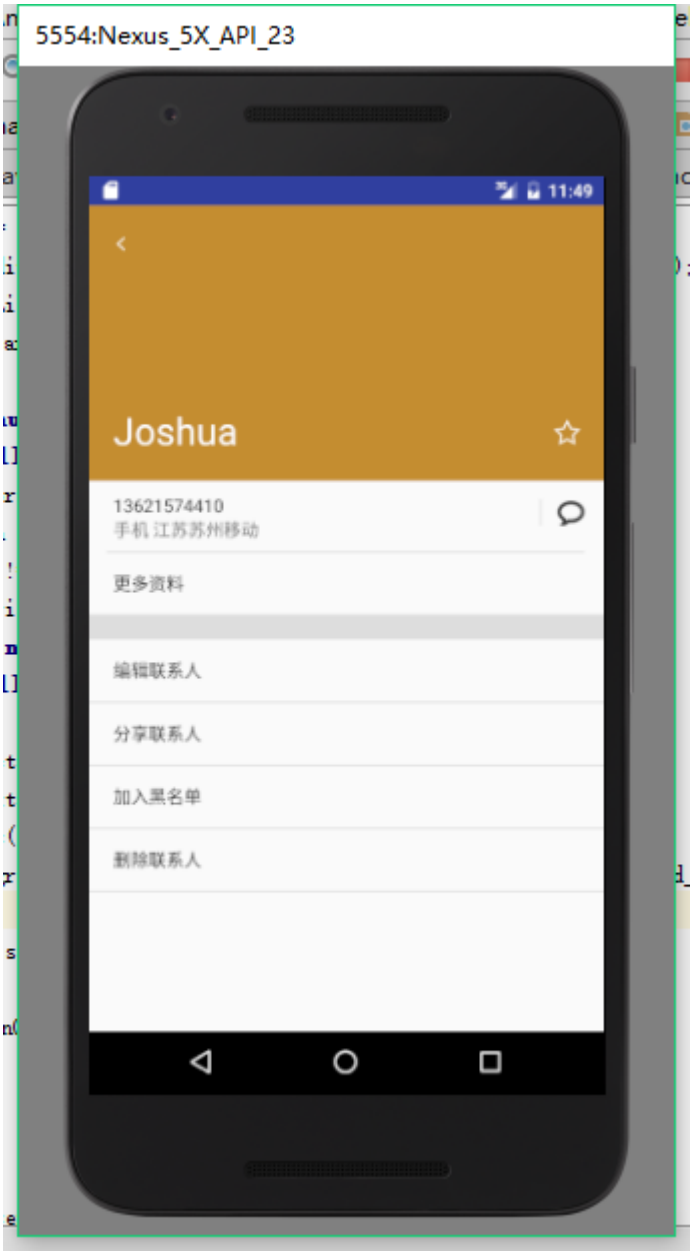
5. 长按任一联系人（Edwin）


5554:Nexus_5X_API_23
11:49
A
Aaron
E
Elvis
删除联系人
确认删除联系人Edwin?
取消
确认
F
Frank
J
Joshua

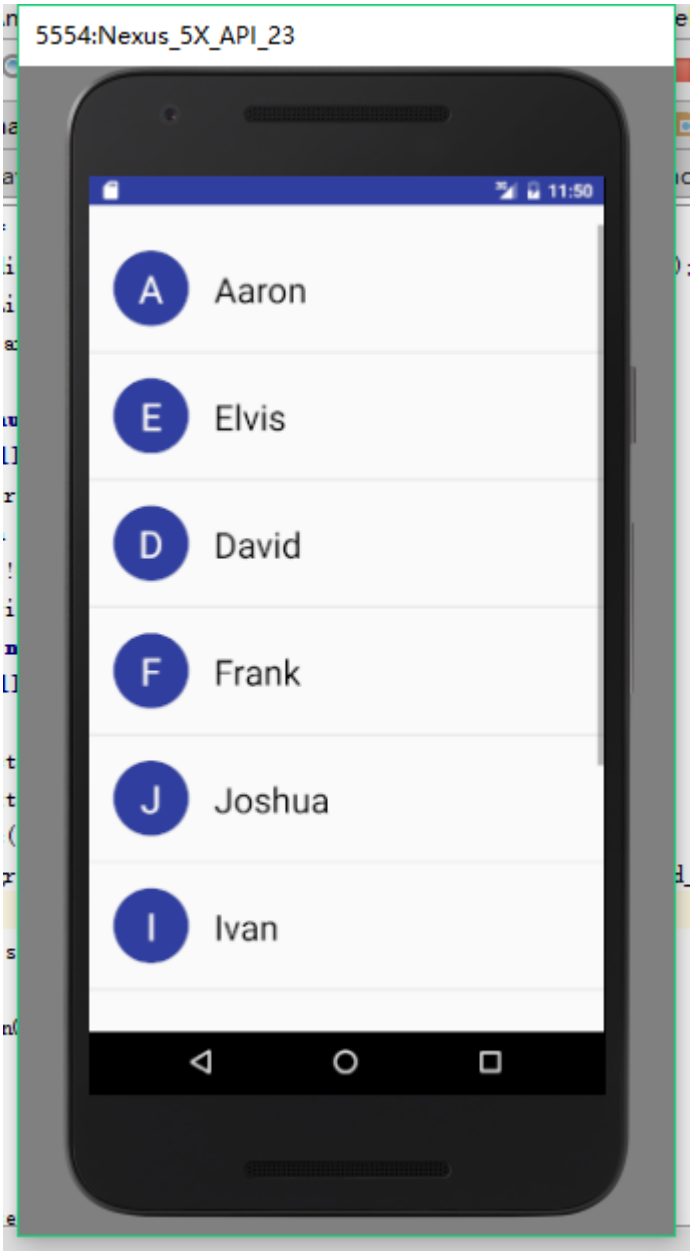
6. 点击确认


5554:Nexus_5X_API_23
11:49
A
Aaron
E
Elvis
D
David
F
Frank
J
Joshua
I
Ivan

7. 再次点击任一联系人



8. 点击返回按钮，Edwin 确实被删除了



四、实验心得

通过此次实验，我懂得了 Intent、Bundle 的使用和 ListView 的应用，熟悉了各种 Layout 特别是 RelativeLayout 和 FrameLayout 的使用，实现了自定义的 Adapter，也掌握了在 Activity 间传递消息的方法，深刻领悟了 UI 设计的不易。

实现 Parcelable 接口可以在切换 Activity 时使用 putParcelable 函数进行传递消息，实现该接口需要实现两个函数

```
@Override
public int describeContents() {
    return 0;
}

@Override
public void writeToParcel(Parcel dest, int flags) {
    dest.writeString(name);
    dest.writeString(phone_number);
    dest.writeString(attribution);
    dest.writeString(background_color);
}

public static final Parcelable.Creator<Contact> CREATOR = new Creator<Contact>() {

    @Override
    public Contact createFromParcel(Parcel source) {
        Contact contact = new Contact();
        contact.name = source.readString();
        contact.phone_number = source.readString();
        contact.attribution = source.readString();
        contact.background_color = source.readString();
        return contact;
    }
```

此外，我还实现了用于可以序列化和反序列化 Contact 数组的函数

```
    //供反序列化本类数组时调用的
    @Override
    public Contact[] newArray(int size) {
        return new Contact[size];
    }
};
```

注意在反序列化函数要按照序列化函数中序列化的顺序一一反序列化各个字段的值

最后，需要注意的一点是使用 Color.parseColor 时需要在颜色前添

加一个 # 号，否则在运行时程序会抛出异常

```
background.setBackgroundColor((Color.parseColor('#' + contact.getBackground_color())));
```