中山大學
SUN YAT-SEN UNIVERSITY

《手机平台应用开发》

数据存储（二）

学 院 名 称 ： 数据科学与计算机学院

成　　员 ： 陈伟宸

时　　间 ： 2016 年 11 月 18 日

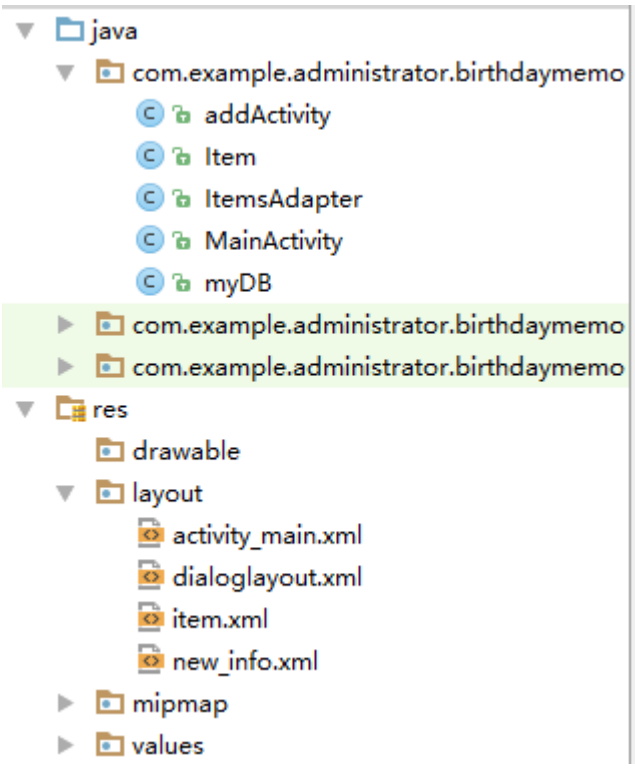
一、实验环境

操作系统：Windows 10

IDE：Android Studio 2.2.2

二、实验过程

1. 新建一个空项目，新建如下文件：



2. 设计 MainActivity 的界面，

```
<Button
    android:id="@+id/add_item"
    android:layout_gravity="center_horizontal"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="增加条目"
    android:background="#228B22"
    android:textColor="#FFFFFF"
    android:textSize="20sp"
    android:layout_marginTop="20dp"/>
```

```
<LinearLayout
    android:layout_marginLeft="30dp"
    android:layout_marginTop="20dp"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <TextView
        android:layout_weight="1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="姓名"
        android:textSize="30sp"
        android:textStyle="bold"/>
    <TextView
        android:layout_weight="2"
        android:textSize="30sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="生日"
        android:textStyle="bold"/>
    <TextView
        android:layout_weight="1"
        android:textSize="30sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="礼物"
        android:textStyle="bold"/>
</LinearLayout>

<View
    android:layout_marginTop="5dp"
    android:layout_marginRight="5dp"
    android:layout_marginLeft="5dp"
    android:layout_width="fill_parent"
    android:layout_height="2px"
    android:background="@android:color/darker_gray"  />

<ListView
    android:id="@+id/items"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginLeft="30dp">
</ListView>
```

3. 设计 addActivity 的界面，

```
<LinearLayout
    android:layout_marginLeft="20dp"
    android:layout_marginRight="20dp"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <TextView
        android:textSize="15sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="姓名: "/>
    <EditText
        android:id="@+id/add_name"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>
</LinearLayout>
<LinearLayout
    android:layout_marginLeft="20dp"
    android:layout_marginRight="20dp"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <TextView
        android:textSize="15sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="生日: "/>
    <EditText
        android:id="@+id/add_birth"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</LinearLayout>
```

```
<LinearLayout
    android:layout_marginLeft="20dp"
    android:layout_marginRight="20dp"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <TextView
        android:textSize="15sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="礼物："/>
    <EditText
        android:id="@+id/add_gift"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</LinearLayout>

<Button
    android:id="@+id/add_add"
    android:layout_gravity="center_horizontal"
    android:background="#228B22"
    android:textColor="#FFFFFF"
    android:textSize="20sp"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="增加"/>
```

4. 实现 Item 类，用于存放姓名、生日、礼物等信息

```java
public class Item {
    private String name;
    private String birth;
    private String gift;
    private int _id;

    public Item() {}
    public Item(int id, String name, String birth, String gift) {
        this._id = id;
        this.name = name;
        this.birth = birth;
        this.gift = gift;
    }
    public Item(String name, String birth, String gift) {
        this.name = name;
        this.birth = birth;
        this.gift = gift;
    }

    public String getName() { return name; };
    public String getBirth() { return birth; };
    public String getGift() { return gift; };
    public int getId() { return _id; }

    public void setName(String name) { this.name = name; };
    public void setBirth(String birth) { this.birth = birth; };
    public void setGift(String gift) { this.gift = gift; }
    public void setId(int id) { this._id = id; }
}
```

5. 实现与数据库交互的 myDB 类，该类继承于 SQLiteOpenHelper 类

（1）重写 onCreate 函数，在数据库中创建表，表的主键为_id，且_id 自动增长

```java
@Override
public void onCreate(SQLiteDatabase sqLiteDatabase) {
    String CREATE_TABLE = "CREATE TABLE if not exists " + TABLE_NAME + " " +
            "(_id INTEGER PRIMARY KEY AUTOINCREMENT," +
            " name TEXT UNIQUE, birth TEXT, gift TEXT)";
    sqLiteDatabase.execSQL(CREATE_TABLE);
}
```

（2）重写 onUpgrade 函数

```
@Override
public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i1) {

}
```

（3）实现 insert 函数，实现插入不同名字的 item

```
public boolean insert(Item item) {
    SQLiteDatabase db = getWritableDatabase();
    SQLiteDatabase check = getReadableDatabase();
    if (check.query(TABLE_NAME, new String[] { "name" }, "name=\"" +
            item.getName() + "'", null, null, null, null).getCount() != 0)
        return false;
    ContentValues cv = new ContentValues();
    cv.put("name", item.getName());
    cv.put("birth", item.getBirth());
    cv.put("gift", item.getGift());
    db.insert(TABLE_NAME, null, cv);
    return true;
}
```

（4）实现 update 函数，更新 item 的信息

```
public long update(Item item) {
    SQLiteDatabase db = getWritableDatabase();
    ContentValues cv = new ContentValues();
    cv.put("birth", item.getBirth());
    cv.put("gift", item.getGift());
    return db.update(TABLE_NAME, cv, "_id=" + item.getId(), null);
}
```

（5）实现 delete 函数，根据 id 删除某个 item

```
public long delete(int id) {
    SQLiteDatabase db = getWritableDatabase();
    return db.delete(TABLE_NAME, "_id=" + id, null);
}
```

（6）实现 ConvertToItem 函数，该函数根据 Cursor 返回该 Cursor 中查询到的所有 item

```
private List<Item> ConvertToItem(Cursor cursor) {
    int resultCounts = cursor.getCount();
    if (resultCounts == 0 || !cursor.moveToFirst())
        return null;
    List<Item> items = new ArrayList<>();
    for (int i = 0; i < resultCounts; i++) {
        items.add(new Item());
        items.get(i).setId(cursor.getInt(0));
        items.get(i).setName(cursor.getString(cursor.getColumnIndex("name")));
        items.get(i).setBirth(cursor.getString(cursor.getColumnIndex("birth")));
        items.get(i).setGift(cursor.getString(cursor.getColumnIndex("gift")));
        cursor.moveToNext();
    }
    return items;
}
```

（7）实现 queryAll 和 query 函数，分别用于查询所有的 item 和根据某个 id 查询某个 item

```
public List<Item> queryAll() {
    SQLiteDatabase db = getReadableDatabase();
    Cursor cursor = db.query(TABLE_NAME, null,
            null, null, null, null, null);
    return ConvertToItem(cursor);
}

public Item query(int id) {
    SQLiteDatabase db = getReadableDatabase();
    Cursor cursor = db.query(TABLE_NAME, null,
            "_id=" + id, null, null, null, null);
    try {
        return ConvertToItem(cursor).get(0);
    } catch (Exception e) {
        return null;
    }
}
```

6. 实现 MainActivity

（1）加载数据库中的 item

```
db = new myDB(getApplicationContext(), DB_NAME, null, DB_VERSION);
items = db.queryAll();
```

（2）实现 ItemsAdapter，用于在 ListView 中显示 item

```
public class ItemsAdapter extends BaseAdapter {
    private List<Item> items;
    private Context context;

    public ItemsAdapter(Context context, List<Item> items) {
        this.context = context;
        this.items = items;
    }

    @Override
    public int getCount() {
        if (items == null)
            return 0;
        return items.size();
    }

    @Override
    public Object getItem(int i) {
        if (items == null)
            return null;
        return items.get(i);
    }

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

```
@Override
public View getView(int i, View view, ViewGroup viewGroup) {
    View convertView;
    ViewHolder viewHolder;

    if (view == null) {
        convertView = LayoutInflater.from(context).inflate(R.layout.item, null);
        viewHolder = new ViewHolder();
        viewHolder.name = (TextView) convertView.findViewById(R.id.name);
        viewHolder.birth = (TextView) convertView.findViewById(R.id.birth);
        viewHolder.gift = (TextView) convertView.findViewById(R.id.gift);
        convertView.setTag(viewHolder);
    }
    else {
        convertView = view;
        viewHolder = (ViewHolder) convertView.getTag();
    }

    viewHolder.name.setText(items.get(i).getName());
    viewHolder.birth.setText(items.get(i).getBirth());
    viewHolder.gift.setText(items.get(i).getGift());


    return convertView;
}

private class ViewHolder {
    public TextView name;
    public TextView birth;
    public TextView gift;
}
```

```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/name"
        android:layout_weight="1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="姓名"
        android:textSize="30sp" />
    <TextView
        android:id="@+id/birth"
        android:layout_weight="2"
        android:textSize="30sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="生日"/>
    <TextView
        android:id="@+id/gift"
        android:layout_weight="1"
        android:textSize="30sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="礼物"/>

</LinearLayout>
```

（3）如果数据库中存在 item，则显示在 MainActivity 中

```
if (items != null) {
    itemsAdapter = new ItemsAdapter(getApplicationContext(), items);
    item_list.setAdapter(itemsAdapter);
}
```

（4）实现点击“增加条目”按钮，跳转到 addActivity 中

```
add.setOnClickListener((view) -> {
        Intent intent = new Intent(MainActivity.this, addActivity.class);
        startActivity(intent);
});
```

（5）设计点击 ListView 中的 item 弹出的自定义对话框界面

```
<TextView
    android:layout_marginTop="5dp"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="o(*￣▽￣*)ブ"
    android:textSize="20sp"
    android:textColor="#1E90FF"
    />
<View
    android:layout_marginTop="10dp"
    android:layout_width="fill_parent"
    android:layout_height="5px"
    android:background="#1E90FF"  />


<LinearLayout
    android:layout_marginTop="15dp"
    android:layout_marginLeft="20dp"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <TextView
        android:textSize="20sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="姓名:  "/>
    <TextView
        android:id="@+id/update_name"
        android:textSize="20sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="老王家"/>
</LinearLayout>
```

```
<LinearLayout
    android:layout_marginLeft="20dp"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <TextView
        android:textSize="20sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="生日:  "/>
    <EditText
        android:id="@+id/update_birth"
        android:textSize="20sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="4.1"/>
</LinearLayout>
```

Item.xml

```
<LinearLayout
    android:layout_marginLeft="20dp"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <TextView
        android:textSize="20sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="礼物:  "/>
    <EditText
        android:id="@+id/update_gift"
        android:textSize="20sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="辣条"/>
</LinearLayout>
```

```
<LinearLayout
    android:layout_marginTop="20dp"
    android:layout_marginLeft="20dp"
    android:layout_marginBottom="20dp"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <TextView
        android:textSize="20sp"
        android:textColor="#228B22"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="电话:  "/>
    <TextView
        android:id="@+id/update_number"
        android:textColor="#228B22"
        android:textSize="20sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text=": 668393"/>
</LinearLayout>
```

（6）实现点击 ListView 中的 item 弹出的自定义对话框

① 加载自定义的对话框界面，并获取到各个控件

```
LayoutInflater factory = LayoutInflater.from(MainActivity.this);
View view1 = factory.inflate(R.layout.dialoglayout, null);
AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);
builder.setView(view1);

name = (TextView) view1.findViewById(R.id.update_name);
birth = (EditText) view1.findViewById(R.id.update_birth);
gift = (EditText) view1.findViewById(R.id.update_gift);
num = (TextView) view1.findViewById(R.id.update_number);

name.setText(items.get(ii).getName());
birth.setText(items.get(ii).getBirth());
gift.setText(items.get(ii).getGift());
```

② 在通讯录中根据点击的 item 的姓名查询电话号码

```
Cursor c = getContentResolver().query(
        ContactsContract.CommonDataKinds.Phone.CONTENT_URI, null,
        ContactsContract.CommonDataKinds.Phone.DISPLAY_NAME + "=\""
                + name.getText().toString() + "'", null, null);
String number = new String();
while (c != null && c.moveToNext()) {
    number += c.getString(c.getColumnIndex(
            ContactsContract.CommonDataKinds.Phone.NUMBER)) + " ";
}
num.setText(number);
c.close();
```

③ 实现点击“确认修改”按钮，更新数据库信息，并通知 UI 和“放弃修改”按钮

```
builder.setPositiveButton("确认修改", (dialogInterface, i) -> {
        db.update(new Item(items.get(ii).getId(), name.getText().toString(),
                birth.getText().toString(), gift.getText().toString()));
        items.get(ii).setName(name.getText().toString());
        items.get(ii).setBirth(birth.getText().toString());
        items.get(ii).setGift(gift.getText().toString());
        itemsAdapter.notifyDataSetChanged();
}).setNegativeButton("放弃修改", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialogInterface, int i) {

    }
}).create().show();
```

（7）实现长按 ListView 中的 item 弹出的删除对话框，确认删除信息并更新数据库

```
item_list.setOnItemLongClickListener((adapterView, view, ii, l) -> {
        AlertDialog.Builder deleteAlert = new AlertDialog.Builder(MainActivity.this);
        deleteAlert.setMessage("是否删除?")
                .setPositiveButton("是", (dialogInterface, i) -> {
                        db.delete(items.get(ii).getId());
                        items.remove(ii);
                        itemsAdapter.notifyDataSetChanged();
                }).setNegativeButton("否", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialogInterface, int i) {

            }
        }).create().show();
        return true;
});
```

7. 实现 addActivity

实现点击“增加”按钮，向数据库插入 item，在插入成功时跳转到 MainActivity 界面，失败时提示失败原因

```
add.setOnClickListener((view) -> {
        if (name.getText().toString().isEmpty()) {
            toast.setText("名字为空，请完善");
            toast.show();
        }
        else {
            boolean success = db.insert(new Item(name.getText().toString(),
                    birth.getText().toString(), gift.getText().toString()));
            if (success) {
                Intent intent = new Intent(addActivity.this, MainActivity.class);
                startActivity(intent);
            }
            else {
                toast.setText("名字重复啦，请核查");
                toast.show();
            }
        }
});
```

三、实验结果

1. 运行程序

2. 点击“增加条目”


Android Emulator - Nexus_5X_API_19:5554
5:20
BirthdayMemo
姓名：
生日：
礼物：
增加

3. 输入姓名，生日，礼物


Android Emulator - Nexus_5X_API_19:5554
5:34
BirthdayMemo
姓名：Ming
生日：11.11
礼物：Nothing
增加

4. 点击“增加”按钮


Android Emulator - Nexus_5X_API_19:5554
5:34
BirthdayMemo
增加条目
姓名
生日
礼物
Ming
11.11
Nothing

5. 点击该 item

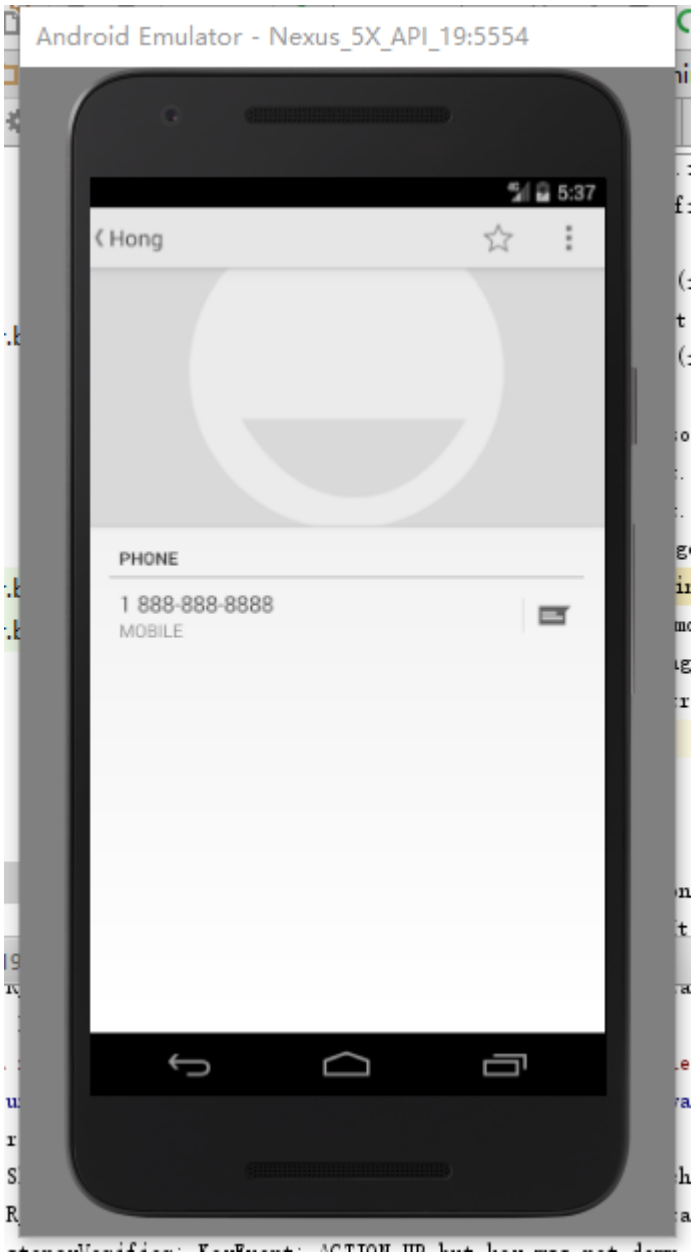
6. 修改礼物为 Cookie，并点击“确认修改”

7. 点击“增加条目”，尝试再次添加姓名为“Ming”的 item

提示名字重复

8. 添加 Hong

9. 在通讯录为 Hong 添加电话号码



10. 回到 app，点击 Hong

成功查询到 Hong 的电话号码

11. 重新运行 app

成功读取 Item

12. 长按 Ming，删除


Android Emulator - Nexus_5X_API_19:5554
6:04
BirthdayMemo
增加条目
姓名
生日
礼物
Ming
11.11
Cookie
Hong
1.1
Nothing
是否删除?
否
是

四、实验心得

1. 在查询时如果使用了选择条件，应该在参数两端加上双引号

```
if (check.query(TABLE_NAME, new String[] { "name" }, "name=\"" +
        item.getName() + "'", null, null, null, null).getCount() != 0)
    return false;
```

```
Cursor c = getContentResolver().query(
        ContactsContract.CommonDataKinds.Phone.CONTENT_URI, null,
        ContactsContract.CommonDataKinds.Phone.DISPLAY_NAME + "=\""
                + name.getText().toString() + "'", null, null);
```

2. 使用自定义的对话框时，获取对话框的空间应该使用自定义的 view 获取

```
name = (TextView) view1.findViewById(R.id.update_name);
birth = (EditText) view1.findViewById(R.id.update_birth);
gift = (EditText) view1.findViewById(R.id.update_gift);
num = (TextView) view1.findViewById(R.id.update_number);
```

3. 各版本的 API 数据库的 insert 函数返回值不同，不能直接使用 insert 的返回值判断插入是否成功

4. 获取联系人时应该判断是否获取成功

```
while (c != null && c.moveToNext()) {
```