

安卓移动应用开发

软件名称：SQLite_ListView

完成时间：2017/12/5

学号：20152100165

姓名：陈玉淋

邮箱：2282095989@qq.com

一、软件内容简介

实现向 SQLite 数据库中批量添加数据，并通过 ListView 显示添加后的数据。

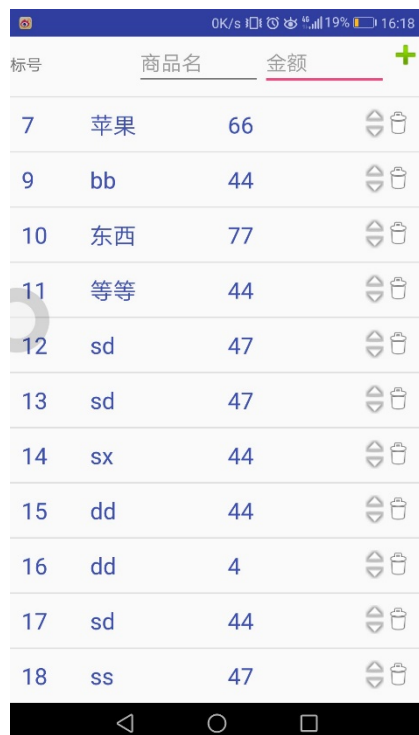
为了实现上述需求，计划开发一个应用 sqlite 和 ListView 的小程序。百度发现此类程序有记事本，通讯录与购物车。双十二接近所以最后选择开发一个购物车。

软件主要功能

1. 将购物车中的商品以列表的形式展示，
2. 对购物车中的商品进行增、删、改、查操作。
3. 使用 ListView 和 SQLite 数据库。

二、界面设计(可拷屏界面)

做下图页面，点击+号，输入商品名称，金额。商品就会显示在下面的 ListView 上，数据是从数据库中获取的。



标号	商品名	金额	
7	苹果	66	↕ 🗑
9	bb	44	↕ 🗑
10	东西	77	↕ 🗑
11	等等	44	↕ 🗑
12	sd	47	↕ 🗑
13	sd	47	↕ 🗑
14	sx	44	↕ 🗑
15	dd	44	↕ 🗑
16	dd	4	↕ 🗑
17	sd	44	↕ 🗑
18	ss	47	↕ 🗑

三、代码设计(可拷屏代码)

1.设计用户交互界面

添加的三个 TextView 分别用于显示数据库中的某条数据 id，商品名称，金额。

添加的三个 ImageView 用于增加金额，减少金额，删除数据。

1.1 (activity_main.xml)

1.1.1 用 ImageView 显示图片

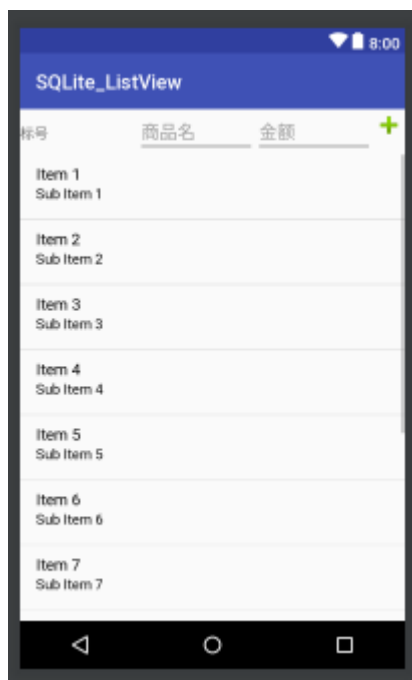
使用 ImageView 的属性 `Android:src` 来指定 ImageView 要显示的图片，但是只显示图片原图大小。

`android:src="@android:drawable/ic_input_add"` 显示图片的原图大小

注意：如果使用 `Android:background` 属性，图片大小会根据 ImageView 大小进行拉伸。

1.1.2 使用 ListView 将购物车中的商品以陈列的形式展示。

1.1.3 效果图



1.1.4 代码

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout

xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
xmlns:app="http://schemas.android.com/apk/res-auto"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical"

tools:context="com.example.chenyulin.sqlite_listview.Ma
inActivity">

    <LinearLayout
        android:id="@+id/addLL"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">
        <TextView
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="标号" />
        <EditText
            android:id="@+id/nameEdT"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:hint="商品名"
            android:inputType="textPersonName" />
        <EditText
            android:id="@+id/balanceET"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:hint="金额"
            android:inputType="number" />
        <ImageView
            android:id="@+id/addIV"
            android:layout_width="wrap_content"
```

```

        android:layout_height="wrap_content"
        android:onClick="add"

        android:src="@android:drawable/ic_input_add"/>

    </LinearLayout>

    <ListView
        android:id="@+id/accountLV"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_below="@id/addLL">

    </ListView>

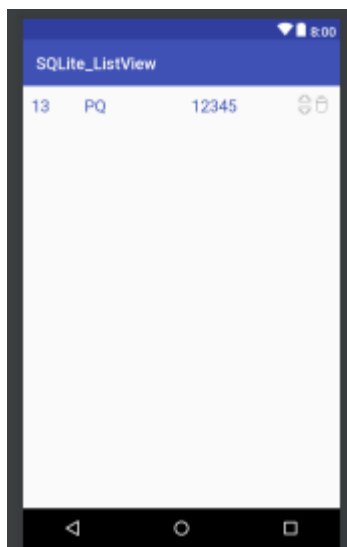
</LinearLayout>

```

1.2 创建 ListView Item 布局

在 res/layout 目录下创建一个 item.xml 文件，创建 ListView Item 布局，添加三个 TextView，分别用于显示数据库中的某条数据的 id、商品名称、金额，三个 ImageView 用于增加金额、减少金额、删除数据。

1.2.1 效果图



1.2.2 代码

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:padding="10dp">

    <TextView
        android:id="@+id/idTV"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="13"
        android:textColor="@color/colorPrimary"
        android:textSize="20sp" />

    <TextView
        android:id="@+id/nameTV"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="2"
        android:singleLine="true"
        android:text="PQ"
        android:textColor="@color/colorPrimary"
        android:textSize="20sp" />

    <TextView
        android:id="@+id/balanceTV"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="2"
        android:singleLine="true"
        android:text="12345"
        android:textColor="@color/colorPrimary"
        android:textSize="20sp" />

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical">
```

```
        <ImageView
            android:id="@+id/upIV"
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:layout_marginBottom="2dp"

android:src="@android:drawable/arrow_up_float" />

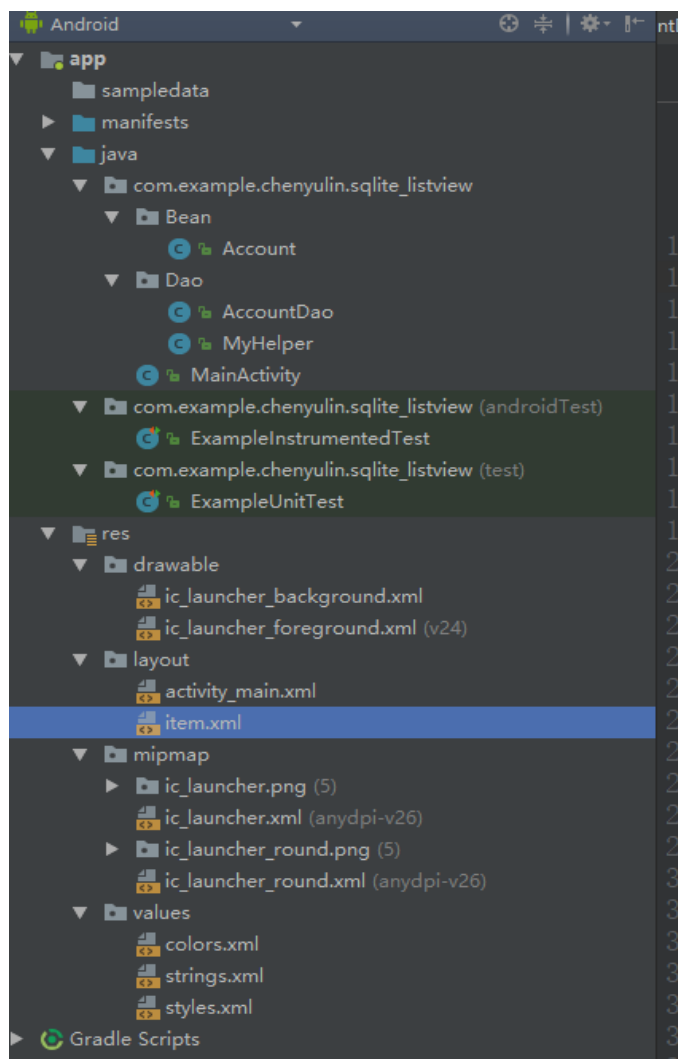
        <ImageView
            android:id="@+id/downIV"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"

android:src="@android:drawable/arrow_down_float" />
    </LinearLayout>

    <ImageView
        android:id="@+id/deleteIV"
        android:layout_width="25dp"
        android:layout_height="25dp"
        android:src="@android:drawable/ic_menu_delete"
    />

</LinearLayout>
```

2. 创建数据库



2.1 新建数据库包

2.1.1 MyHelper 类

创建数据库属于数据操作，因此需要创建一个名为 dao 的包。并在该包下定义一个 MyHelper 类继承自 SQLiteOpenHelper。

2.1.2 MyHelper 代码

```
package com.example.chenyulin.sqlite_listview.Dao;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;
```



```

/**
 * Created by ChenYulin on 2017/12/4.
 */

public class MyHelper extends SQLiteOpenHelper{
    //由于父类没有无参构造参数，所以子类必须指定调用父
    //类有参的构造函数
    public MyHelper(Context context) {
        super(context, "product.db", null, 2);
    }
    @Override
    public void onCreate(SQLiteDatabase db) {
        Log.d("MyHelper", "OnCreate");
        db.execSQL("create table account(_id integer
primary key autoincrement, name varchar(20), balance
integer)");
        //id 主键，商品名称列，金额列

    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int
oldVersion, int newVersion) {
        Log.d("MyHelper", "OnUpgrade");
    }
}

```

2.2 Javabeans 类

在操作数据库时将数据存放至一个 JavaBean 对象中操作起来会比较方便。因此，需要创建一个 bean 包用于存放 Javabeans 类，在包中定义一个类 Account，

具体代码如下：

```

package com.example.chenyulin.sqlite_listview.Bean;

/**
 * Created by ChenYulin on 2017/12/4.
 */

```

```
public class Account {
    private Long id;
    private String name;
    private Integer balance;

    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public Integer getBalance() {
        return balance;
    }

    public void setBalance(Integer balance) {
        if(balance >= 0)
            this.balance = balance;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public Account(long id, String name, Integer
balance) {
        super();
        this.balance = balance;
        this.name = name;
        this.id = id;
    }

    public Account(String name, Integer balance) {
        super();
        this.name = name;
        this.balance = balance;
    }
}
```

```

    }
    public Account()
    {
        super();
    }
    public String toString()
    {
        return "[序号: "+id+", 商品名称"+name+", 余额: "+balance+"]";
    }
}

```

2.3 创建数据操作逻辑类

编写数据逻辑操作类，在 dao 包下创建一个 AccountDao 类用于操作数据，该类创建对数据进行增、删、改、查操作的方法。

需要注意，在 insert () 方法中调用了 db.insert () 方法，这个方法第二个参数如果传入 null，是无法插入一条空数据的。如果想插入一条空数据，第二个参数必须写一个列名（任意列），传入的这个列名是用来拼接 SQL 语句的，例如，INSERT INTO account (null) VALUES (null)。

2.3.1 代码

```

package com.example.chenyulin.sqlite_listview.Dao;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;

import com.example.chenyulin.sqlite_listview.Bean.Account;

import java.util.ArrayList;
import java.util.List;

/**

```

** Created by ChenYulin on 2017/12/4.*

**/*

```
public class AccountDao {
    private MyHelper helper;
    public AccountDao(Context context) {
        //创建 Dao 时，创建 helper
        helper=new MyHelper(context);
    }
    //-----插入
    public void insert(Account account){
        //读取数据库对象
        SQLiteDatabase db=helper.getWritableDatabase();
        //用来装载要插入的数据的 MapM<列名， 列的值>
        ContentValues values=new ContentValues();
        values.put("name",account.getName());
        values.put("balance",account.getBalance());
        //向 account 表插入数据 values
        long id=db.insert("account",null,values);
        account.setId(id); //得到 id
        db.close(); //关闭数据库
    }
    //-----删除， 根据 id 删除
    public int delete(long id)
    {
        SQLiteDatabase db=helper.getWritableDatabase();
        //按条件删除指定表中的数据， 返回受影响的行数
        int count=db.delete("account","_id=?",new
String[] {id+""});
        db.close();
        return count;
    }
    //-----更新数据
    public int update(Account account)
    {
        SQLiteDatabase db=helper.getWritableDatabase();
        ContentValues values=new ContentValues(); //要
修改的数据 键值对
        values.put("name",account.getName());
        values.put("balance",account.getBalance());
        int
count=db.update("account",values,"_id=?",new
String[] { account.getId()+""}); //更新并得到行数
    }
}
```

```

        db.close();
        return count;
    }
    //-----查询所有的数据倒序排列
    public List<Account> queryAll()
    {
        SQLiteDatabase db=helper.getWritableDatabase();
        Cursor
c=db.query("account",null,null,null,null,null,"balance
DESC");
        List<Account> list=new ArrayList<Account>();
        while(c.moveToNext())
        {
            //可以根据列名获取索引
            long id=c.getLong(c.getColumnIndex("_id"));
            String name=c.getString(1);
            int balance=c.getInt(2);
            list.add(new Account(id,name,balance));

        }
        c.close();
        db.close();
        return list;
    }
}

```

3. 编写界面交互代码（MainActivity）

数据库的操作完成后需要界面与数据库进行交互，用于实现将数据库中的数据以 ListView 的形式展示在界面上。

重要知识点：

(1)ListView 的 setOnItemClickListener () 方法：该方法用于监听 Item 的点击事件，在使用该方法时需要传入一个 OnItemClickListener 的实现类对象，并且需要实现 onItemClick 方法。当点击 ListView 的 Item 时就会触发 Item 的点击事件然后会回调 onItemClick () 方法。

(2)ListView 的 `setSelection ()` 方法：该方法的作用是设置当前选中的条目。假设当前屏幕一屏只能显示 10 条数据，当添加第 11 条数据时，调用此方法就会将第 11 条数据显示在屏幕上，将第一条数据滑出屏幕外。

(3)Adapter 的 `notifyDataSetChanged ()` 方法：该方法用于重新数据，当数据适配器中的内容发生变化时，会调用此方法，重新执行 `BaseAdapter` 中的 `getView ()` 方法。

具体代码如下：

```
package com.example.chenyulin.sqlite_listview;

import android.app.Activity;
import android.app.AlertDialog.Builder;
import android.content.DialogInterface;
import android.os.Bundle;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.BaseAdapter;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

import java.util.List;

import com.example.chenyulin.sqlite_listview.Bean.Account;
import com.example.chenyulin.sqlite_listview.Dao.AccountDao;

public class MainActivity extends Activity {
    //需要适配的数据集合
    private List<Account> list;
```

```

//需要增删改查操作类
private AccountDao dao;
//输入姓名的 edittext
private EditText nameET;
//输入金额的 edittext
private EditText balanceET;
//适配器
private MyAdapter adapter;
//Listview
private ListView accountLV;

public MainActivity() {
}

@Override
protected void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    //初始化控件
    initView();
    dao=new AccountDao(this);
    //从数据库查询所有数据
    list=dao.queryAll();
    adapter=new MyAdapter();
    accountLV.setAdapter(adapter);//给 ListView 添加
适配器（自动把数据生成条目）
}
//初始化控件
private void initView()
{
accountLV=(ListView)findViewById(R.id.accountLV);
    nameET=(EditText) findViewById(R.id.nameEdT);

balanceET=(EditText)findViewById(R.id.balanceET);
    //添加监听器，监听条目点击事件
    accountLV.setOnItemClickListener(new
MyOnItemClickListener());
}
//activity_main.xml 对应的 inageview 的点击事件触发
的方法
public void add(View v)

```

```

    {
        String name=nameET.getText().toString().trim();
        String
balance=balanceET.getText().toString().trim();
        if("".equals(name) || "".equals(balance))
        {
            Toast.makeText(getApplicationContext(),"商
品名称和金额不能为空",Toast.LENGTH_LONG).show();
            return;
        }
        //三目运算 balance.equals("") 则等于 0
        //如果 balance 不是空字符串，则进行强制类型转换
        Account a=new Account(name,
balance.equals("")?0:Integer.parseInt(balance));
        dao.insert(a); //插入数据库
        list.add(a); //插入集合
        adapter.notifyDataSetChanged(); //刷新界面
        //选中最后一个
        accountLV.setSelection(accountLV.getCount()-1);
        nameET.setText("");
        balanceET.setText("");
    }
    //自定义一个适配器（把数据装到 ListView 的工具）
    private class MyAdapter extends BaseAdapter{
        public int getCount() { //获取条目总数
            return list.size();
        }
        public Object getItem(int position) //根据位置
获取对象
        {
            return list.get(position);
        }
        public long getItemId(int position) //根据位置
获取 id
        {
            return position;
        }
        //获取一个条目视图
        public View getView(int position, View
convertView, ViewGroup parent)
        {
            //重用 convertView
            View

```



```

item=convertView!=null?convertView:View.inflate(getApplicationContext(),R.layout.item,null);
    //获取该视图中的 textview
    TextView idTV=(TextView)
item.findViewById(R.id.idTV);
    TextView nameTV=(TextView)
item.findViewById(R.id.nameTV);
    TextView balanceTV=(TextView)
item.findViewById(R.id.balanceTV);
    //根据当前位置获取 Account 对象
    final Account a=list.get(position);
    //把 Account 对象中的数据放到 textview 中
    idTV.setText(a.getId()+"");
    nameTV.setText(a.getName());
    balanceTV.setText(a.getBalance()+"");
    ImageView upTV=(ImageView)
item.findViewById(R.id.upIV);
    ImageView downTV=(ImageView)
item.findViewById(R.id.downIV);
    ImageView deleteTV=(ImageView)
item.findViewById(R.id.deleteIV);
    //向上箭头的事件触发方法
    upTV.setOnClickListener(new
View.OnClickListener() {
        @Override
        public void onClick(View v) {
            a.setBalance(a.getBalance()+1);
            notifyDataSetChanged();
            dao.update(a);
        }
    });
    //向下箭头的事件触发方法
    upTV.setOnClickListener(new
View.OnClickListener() {
        @Override
        public void onClick(View v) {
            a.setBalance(a.getBalance()-1);
            notifyDataSetChanged();
            dao.update(a);
        }
    });
    //删除图片的点击事件触发的方法
    deleteTV.setOnClickListener(new

```

```

View.OnClickListener() {
    @Override
    public void onClick(View v) {
        //删除数据之前首先弹出一个对话框

        android.content.DialogInterface.OnClickListener
        listener=

            new
        android.content.DialogInterface.OnClickListener() {
            public void
            onClick(DialogInterface dialog, int which) {
                list.remove(a); //从
                集合删除

                dao.delete(a.getId()); //从数据库删除

                notifyDataSetChanged(); //刷新界面
            }
        };
        //创建对话框
        Builder builder =new
        Builder(MainActivity.this);
        builder.setTitle("确定要删除吗? ");
        //设置标题

        //设置确定按钮的文本以及监听器
        builder.setPositiveButton("确定
        ", listener);

        builder.setNegativeButton("取消
        ", null);

        builder.show();
    }
});
return item;
}
}
//Listview 的 item 点击事件
private class MyOnItemClickListener implements
OnItemClickListener{

    public void onItemClick(AdapterView<?>
parent, View view, int position, long id)
    {
        //获取点击位置上的数据
    }
}

```

```

        Account a=(Account)
parent.getItemAtPosition(position);

Toast.makeText(getApplicationContext(),a.toString(),Toa
st.LENGTH_SHORT).show();
    }
}
}

```

四、软件操作流程

运行程序时首先添加商品名称以及金额，点击加号就会将 Edittext 中输入的内容添加至数据库并显示适配到 ListView 中。多添加几条数据，然后点击某条数据的向上的按钮，金额就会增加。向下按钮同理。点击某条数据的删除图案，会弹出一个对话框，确认后就能删除数据了。单击某条数据室，会弹出 toast。

标号	商品名	金额	+	标号	商品名	金额	+
2	橘子	65	⇅🗑	2	橘子	65	⇅🗑
6	xx	44	⇅🗑	6	xx	44	⇅🗑
7	苹果	66	⇅🗑	7	苹果	66	⇅🗑
				8	橘子	32	⇅🗑



五、难点（或遇到的问题）和解决方案

难点就是显示数据库序号时，当你添加 1xx, 2xx, 3xx, 如何删除 3xx 时下次添加的信息就是 4xx 了不美观。

看记事本代码的时候也有这种情况。还没时间解决抱歉。

六、不足之处和今后的设想。

把我的难点解决了。做个仿真商家上传购物车。

本文是购物车程序百度教程的终极结合版，本来想做别的功能，结果。。。。。不过百度的智慧是无穷的，纵观这次过程。有个网友说得好，新手的代码乱拼，高手的代码太简洁。所以我只有所有到看，弃其糟粕，我也算了学会了新技能。

七、参考

http://blog.csdn.net/winnie_hu/article/details/70254380

<http://blog.csdn.net/shasha1021/article/details/70477700>

<http://blog.csdn.net/ly1012053441/article/details/70537132>