中山大学移动信息工程学院本科生实验报告

（2017 年秋季学期）

课程名称：移动应用开发　　任课教师： 郑贵锋

年级	大三	专业（方向）	移动互联网
学号	15352408	姓名	张镓伟
电话	13531810182	Email	709075442@qq.com
开始日期	2017.10.21	完成日期	2017.10.22

一、 实验题目

事件处理 Intent、Bundle 的使用以及 RecyclerView、ListView 的应用

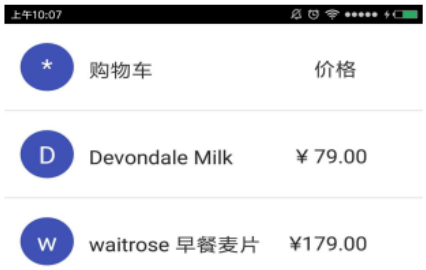
二、 实验目的

1. 复习事件处理
2. 学习 Intent、Bundle 在 Activity 跳转中的应用
3. 学习 RecyclerView、ListView 以及各类适配器的用法

三、 实验内容

本次实验模拟实现一个商品表，有两个界面，第一个界面用于呈现商品，如下所示：

点击右下方的悬浮按钮可以切换到购物车：



上面两个列表点击任意一项后，可以看到详细的信息：

实验要求:布局方面的要求：

1.商品表界面

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。

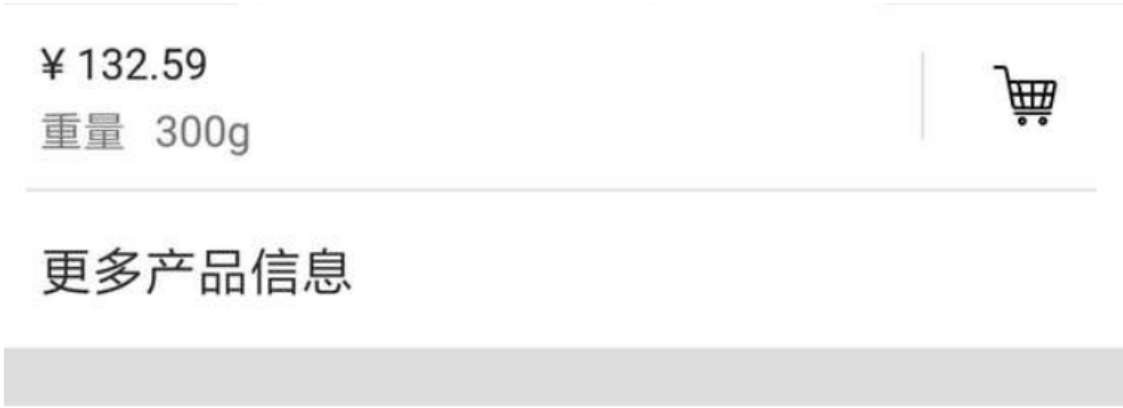
2. 购物车列表界面

在商品表界面的基础上增加一个价格，价格为黑色。

3.商品详情界面顶部

顶部占整个界面的 1/3 。每个商品的图片在商品数据中已给出，图片与这块 view 等高。返回图标处于这块 view 的左上角，商品名字处于左下角，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。这一块建议用 RelativeLayout 实现，以便熟悉 RelativeLayout 的使用。

4.商品详情界面中部



使用的黑色 argb 编码值为#D5000000，稍微偏灰色一点的“重量”、“300g”的 argb 编码值为#8A000000。注意，价格那一栏的下边有一条分割线，argb 编码值为#1E000000，右边购物车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。

5. 商品详情页面底部

底部这里要求使用 ListView 实现

6. 特别提醒，这次的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.使用 RecyclerView 实现商品列表，点击商品列表中的某一个商品会跳转到该商品详情界面，呈现该商品的详细信息；长按商品列表中的第 i 个商品会删除该商品，并且弹出 Toast 提示“移除第 i 个商品”。

2.点击右下方的 FloatingActionButton，从商品列表切换到购物车或从购物车切换到商品列表，并且 FloatingActionButton 的图片要做相应改变。可通过设置 RecyclerView 不可见，ListView 可见来实现从商品列表切换到购物车。可通过设置 RecyclerView 可见，ListView 不可见来实现从购物车切换到商品列表。

3.使用 ListView 实现购物车。点击购物车的某一个商品会跳转到商品详情界面，呈现该商品的详细信息；长按购物中的商品会弹出对话框询问是否移除该商品，点击确定则移除该商品，点击取消则对话框消失。

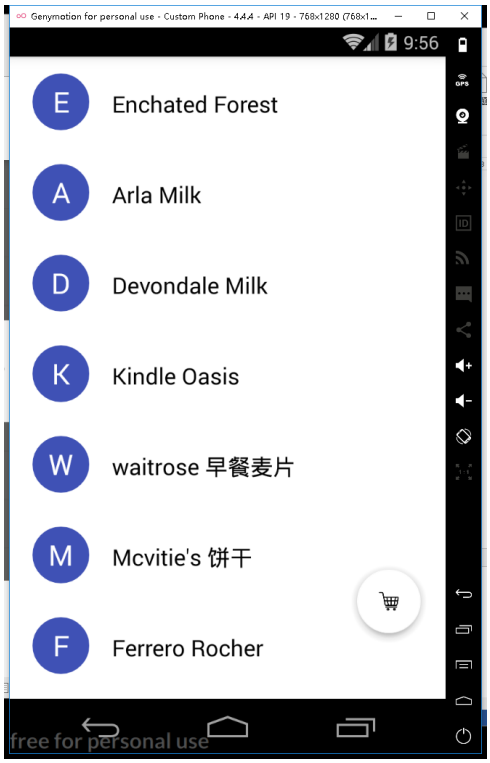
注意对话框中的商品名为被长按的商品

4.商品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。点击购物车图标会将该商品添加到购物车中并弹出 Toast 提示“商品已添加到购物车”。
注：不要求判断购物车是否已有该商品，即如果有一件该商品，添加之后显示两个即可。未退出商品详细界面时，点击多次购物车图标可以只添加一件商品，也可以添加多件到购物车中。

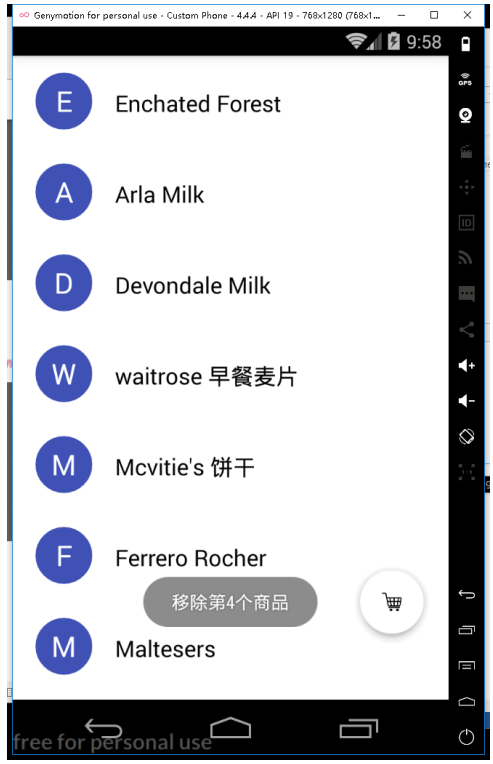
四、 课堂实验结果

（1） 实验截图

商品列表



长按删除 Kindle Oasis



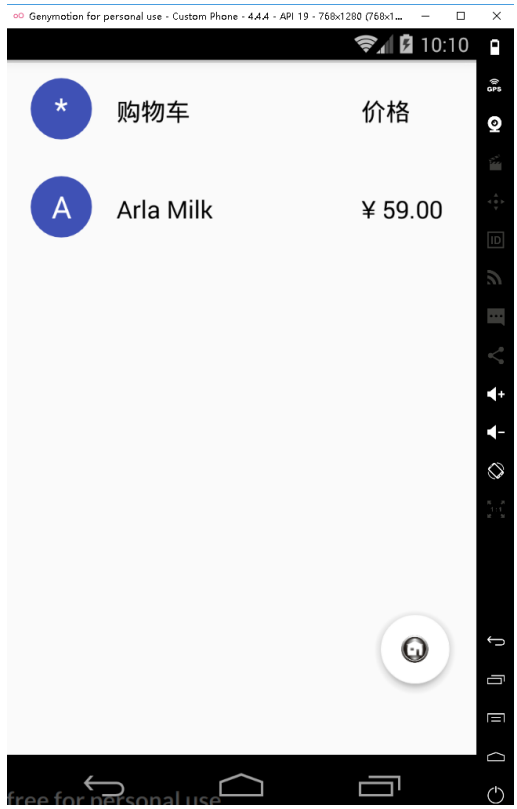
单击进入详情页

点击收藏星星

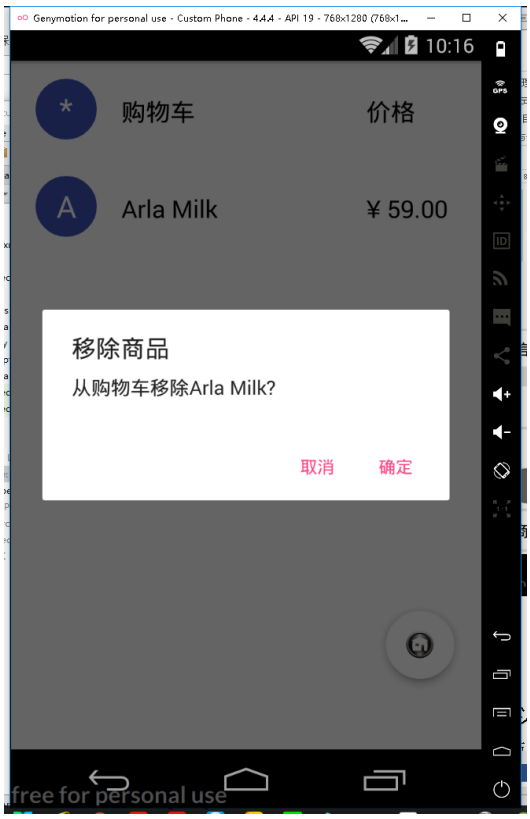
加入购物车

购物车界面

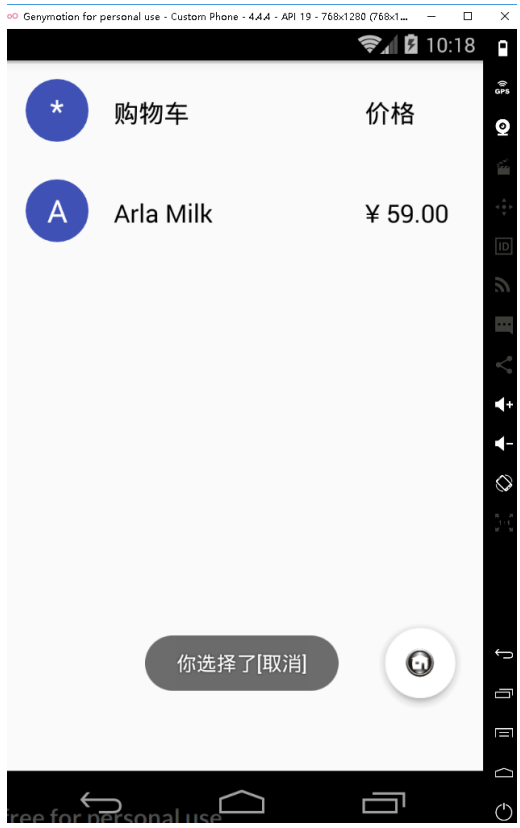
购物车界面长按删除

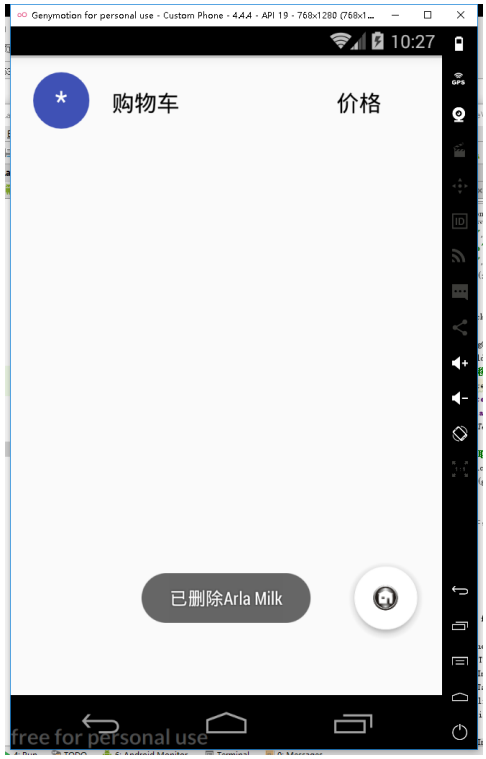


点击取消



点击确定删除后





（2） 实验步骤以及关键代码

1.商品列表布局

```
<android.support.v7.widget.RecyclerView
    android:id="@+id/itemslist"
    android:background="@color/white"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

2. 商品条目样式

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:background="@color/white"
    android:layout_height="80dp">
    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="80dp">
        <TextView
            android:id="@+id/itemlist_icon"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="20dp"
            android:layout_marginTop="15dp"
            android:gravity="center_horizontal|center_vertical"
            android:background="@drawable/round"
            android:text="A"
            android:textSize="25dp"
            android:textColor="@color/white" />

        <TextView
            android:id="@+id/itemlist_name"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center_vertical"
            android:layout_alignBaseline="@+id/itemlist_icon"
            android:layout_toRightOf="@+id/itemlist_icon"
            android:layout_marginLeft="20dp"
            android:text="商品"
            android:textSize="20dp"
            android:textColor="@color/black"/>
    </RelativeLayout>

</LinearLayout>
```

LinearLayout 为水平的; 左边为一个 TextView 是蓝底白字圆圈，右边一个 TextView 显示商品名字。

3. 商品详情上部，通过设置整个页面 weightsum　　商品详情中部为 6，而详情页上部 weight=2 来使其占 1/3

```
<RelativeLayout
    android:id="@+id/itemdetail_r1"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="2"
    android:background="@color/grey">
    <ImageView
        android:id="@+id/itemdetail_r1_picture"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_centerHorizontal="true"
        android:scaleType="fitXY"
        android:src="@drawable/ferrero" />
    <ImageButton
        android:id="@+id/itemdetail_r1_backbutton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginLeft="10dp"
        android:layout_marginTop="10dp"
        android:src="@drawable/back" />
    <ImageButton
        android:id="@+id/itemdetail_r1_starbutton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:layout_marginBottom="10dp"
        android:layout_marginRight="10dp"
        android:src="@drawable/empty_star" />
    <TextView
        android:id="@+id/itemdetail_r1_item_name"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentLeft="true"
        android:layout_marginBottom="15dp"
        android:layout_marginLeft="20dp"
        android:text="Ferrero Rocher"
        android:textColor="@color/black"
        android:textSize="18dp" />
</RelativeLayout>
```

```
<RelativeLayout
    android:id="@+id/itemdetail_r2"
    android:layout_width="fill_parent"
    android:layout_height="0dp"
    android:layout_weight="1">
    <TextView
        android:id="@+id/itemdetail_r2_price"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginLeft="15dp"
        android:layout_marginTop="10dp"
        android:text="价格"
        android:textSize="18dp"
        android:textColor="@color/detailprice"/>
    <TextView
        android:id="@+id/itemdetail_r2_info"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentLeft="true"
        android:layout_marginBottom="10dp"
        android:layout_marginLeft="15dp"
        android:text="信息"
        android:textSize="18dp"
        android:textColor="@color/detailweight"/>
    <ImageView
        android:layout_width="5px"
        android:layout_height="100px"
        android:layout_alignParentRight="true"
        android:layout_centerVertical="true"
        android:layout_marginRight="100dp"
        android:background="@color/cutline" />
    <ImageButton
        android:id="@+id/itemdetail_r2_shopcar"
        android:background="@drawable/shoplist"
        android:layout_width="100px"
        android:layout_height="100px"
        android:layout_alignParentRight="true"
        android:layout_centerVertical="true"
        android:layout_marginRight="20dp" />
</RelativeLayout>
```

更多产品信息

```
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="0dp"
    android:layout_weight="0.8"
    android:orientation="vertical">

    <ImageView
        android:layout_width="match_parent"
        android:layout_height="5px"
        android:background="@color/cutline" />

    <Button
        android:id="@+id/itemdetail_informatic
        android:layout_width="match_parent"
        android:layout_height="45dp"
        android:gravity="center|left"
        android:background="@color/white"
        android:paddingLeft="20dp"
        android:text="更多产品信息"
        android:textColor="@color/black"
        android:textSize="20dp" />

    <ImageView
        android:layout_width="match_parent"
        android:layout_height="25dp"
        android:background="@color/cutline"/>
</LinearLayout>
```

商品详情底部功能布局

```
<ListView
    android:id="@+id/itemdetail_option"
    android:layout_width="wrap_content"
    android:layout_height="0dp"
    android:layout_weight="2.2"></ListView>
```

4.功能条目样式

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent">
    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="48dp">
        <TextView
            android:id="@+id/option"
            android:layout_width="match_parent"
            android:layout_height="35dp"
            android:layout_centerVertical="true"
            android:gravity="center|left"
            android:paddingLeft="20dp"
            android:text="一键下单"
            android:textColor="@color/black"
            android:textSize="20dp" />

    </RelativeLayout>

    <ImageView
        android:layout_width="match_parent"
        android:layout_height="5px"
        android:background="@color/cutline" />

</LinearLayout>
```

5. 购物车条目样式

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="80dp">
    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="80dp">

        <TextView
            android:id="@+id/shopcar_icon"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="20dp"
            android:layout_marginTop="15dp"
            android:background="@drawable/round"
            android:gravity="center_horizontal|center_vertical"
            android:text="*"
            android:textSize="25dp"
            android:textColor="@color/white" />

        <TextView
            android:id="@+id/shopcar_name"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignBaseline="@+id/shopcar_icon"
            android:layout_toRightOf="@+id/shopcar_icon"
            android:layout_marginLeft="20dp"
            android:text="购物车"
            android:textColor="@color/black"
            android:textSize="20dp" />

        <TextView
            android:id="@+id/shopcar_value"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignBaseline="@+id/shopcar_icon"
            android:layout_toRightOf="@+id/shopcar_icon"
            android:layout_marginLeft="220dp"
            android:text="价格"
            android:textColor="@color/black"
            android:textSize="20dp"/>
    </RelativeLayout>
</LinearLayout>
```

JAVA 功能代码部分：

1.首先定义一下商品的信息结构，分别有名字 name，加个 price，信息 info

```java
public class Items {
    private String name;
    private String info;
    private String price;
    public Items(String name,String info,String price){
        this.name=name;
        this.info=info;
        this.price=price;
    }
    public String getName(){return name;}
    public String getInfo(){return info;}
    public String getPrice(){
        return price;
    }
}
```

2.自定商品列表的 RecyclerView.Adapter

```java
public class ItemslistAdapter extends RecyclerView.Adapter<ItemslistAdapter.mViewHolder>{
    public interface OnItemClickListener{
        void onClick(int position);
        void onLongClick(int position);
    }
    private List<Items> items;
    private Context context;
    private OnItemClickListener mOnItemClickListener=null;
    //设置一个监听器
    public void setOnItemClickListener(OnItemClickListener onItemClickListener){
        this.mOnItemClickListener=onItemClickListener;
    }
    //自定义ViewHolder
    class mViewHolder extends RecyclerView.ViewHolder{
        TextView itemlist_name;
        TextView itemlist_icon;
        public mViewHolder(View view){
            super(view);
            itemlist_name=(TextView)view.findViewById(R.id.itemlist_name);
            itemlist_icon=(TextView)view.findViewById(R.id.itemlist_icon);
        }
    }
    //构造函数
    public ItemslistAdapter(List<Items> items, Context context){
        this.items=items;
        this.context=context;
    }
    //创建item视图，并返回相应的viewholder
    @Override
    public mViewHolder onCreateViewHolder(ViewGroup parent, int viewType){
        View view= LayoutInflater.from(context).inflate(R.layout.item,parent,false);
        mViewHolder holder=new mViewHolder(view);
        return holder;
    }
    //绑定数据到正确的item视图
    @Override
    public void onBindViewHolder(final mViewHolder holder, int position){
        holder.itemlist_name.setText(items.get(position).getName());
        holder.itemlist_icon.setText(items.get(position).getName().substring(0,1).toUpperCase());
        if(mOnItemClickListener!=null){
            holder.itemView.setOnClickListener((v) -> {
                    mOnItemClickListener.onClick(holder.getAdapterPosition());
            });
            holder.itemView.setOnLongClickListener((v) -> {
                    mOnItemClickListener.onLongClick(holder.getAdapterPosition());
                    return false;
            });
        }
    }
    //告诉Adapter列表Items的总数
    @Override
    public int getItemCount(){return items.size();}
}
```

这个自定义 Adapter 就是照着实验文档依样画瓢，我们需要 viewholder 里的 view 对象的信息改成我们要显示的信息。同时由于 RecyclerView 没有 OnItemClikListener 方法，需要在适配器中实现，方法是设置一个监听器，当 itemView 被点击的时候，调用该监听器并且将 itemView 的 position 作为参数传递进去

3.初始化商品列表，将商品 item 添加到一个 List 中，这个 List 作为一个参数传到 Adapter 中，同时要设置单击和长按的事件分别为跳转到详情页和删除商品

```
//初始化商品列表
items = new ArrayList<Items>();
for (int i = 0; i < Name.length; i++)
    items.add(new Items(Name[i], Info[i], Price[i]));
itemslistAdapter = new ItemslistAdapter(items, MainActivity.this);
Itemlist.setAdapter(itemslistAdapter);
itemslistAdapter.setOnItemClickListener(new ItemslistAdapter.OnItemClickListener() {
    @Override
    public void onClick(int position) {
        Intent intent = new Intent(MainActivity.this, ItemsDetails.class);
        //使用Intent传递需要用到的值
        intent.putExtra("Name", items.get(position).getName());
        intent.putExtra("Price", items.get(position).getPrice());
        intent.putExtra("Info", items.get(position).getInfo());
        startActivityForResult(intent,1);
        //使用startActivityForResult（Intent intent , intrequestcode）方法打开新的activity,
        // 我们需要为该方法传递一个请求码。请求码的值是根据业务需要由自己设定
    }
    @Override
    public void onLongClick(int position) {
        items.remove(position);
        Toast.makeText(getApplicationContext(),"移除第"+ String.valueOf(position+1 ) + "个商品", Toast.LENGTH_SHORT).show();
        itemslistAdapter.notifyDataSetChanged();
    }
});
```

4.初始化购物车列表，基本同初始化商品列表，但是由于使用的是 ListView，在点击事件的写法上略有不同。

```
//初始化购物车列表
shopcaritems = new ArrayList<Items>();
shopcarAdapter = new ShopcarAdapter(shopcaritems, MainActivity.this);
Shopcarlist=(ListView) findViewById(R.id.shopcarlist_item);
Shopcarlist.setAdapter(shopcarAdapter);
Shopcarlist.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> adapterView, View view, final int i, long l) {
        //i值这一项在列表中的位置，l指的是这一项的id，在ArrayAdapter和SimpleAdapter中，i和l是
        //相等的，在CursorAdapter，l指的是从数据库中取出的数据在数据库中的id值
        Intent intent = new Intent(MainActivity.this, ItemsDetails.class);
        intent.putExtra("Name", shopcaritems.get(i).getName());
        intent.putExtra("Price", shopcaritems.get(i).getPrice());
        intent.putExtra("Info", shopcaritems.get(i).getInfo());
        startActivityForResult(intent,1);
    }
});
Shopcarlist.setOnItemLongClickListener(new AdapterView.OnItemLongClickListener() {
    @Override
    public boolean onItemLongClick(AdapterView<?> adapterView,View view, final int i,long l) {
        final AlertDialog.Builder alertDialog = new AlertDialog.Builder(MainActivity.this);
        alertDialog.setTitle("移除商品").setMessage("从购物车移除" + shopcaritems.get(i).getName() + "?").setPositiveButton("确定", (dialogInterface, j) -> {
                final String removename = new String(shopcaritems.get(i).getName());
                if (shopcaritems.remove(i)!=null ) {
                    shopcarAdapter.notifyDataSetChanged();
                    Toast.makeText(getApplicationContext(), "已删除"+ removename, Toast.LENGTH_SHORT).show();
                }
        }).setNegativeButton("取消", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int j) {
                Toast.makeText(getApplicationContext(), "你选择了[取消]", Toast.LENGTH_SHORT).show();
            }
        }).show();
        //返回true,响应长按事件，返回false 长按短按事件都响应
        return true;
    }

});
```

5.点击浮动按钮切换商品列表和购物车列表的时间，比较简单。就是先设置一个 Tag 记录当前状态，点一下 Tag 在 0 和 1 之间切换，根据 Tag 的状态设置哪个列表可见，哪个列表不可见。

```
    //切换商品列表和购物车列表
    switchbutton = (ImageButton) findViewById(R.id.shopcar);
    switchbutton.setTag("0");
    switchbutton.setOnClickListener((view) -> {
            if (switchbutton.getTag() == "0") {
                switchbutton.setImageResource(R.drawable.mainpage);
                switchbutton.setTag("1");
                Shopcar.setVisibility(View.VISIBLE);
                Itemlist.setVisibility(View.INVISIBLE);
            } else {
                switchbutton.setImageResource(R.drawable.shoplist);
                switchbutton.setTag("0");
                Shopcar.setVisibility(View.INVISIBLE);
                Itemlist.setVisibility(View.VISIBLE);
            }
    });
}
```

6.切换了 activity 再切换回来之后根据传回来的数据的处理，判断是否要网购物车列表中加商品，加的话加多少个。

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent intent) {
    //根据请求码做出相应的业务处理
    if(requestCode==1&&resultCode==1) {
        Bundle bud = intent.getExtras();
        String name = bud.getString("name");
        String price = bud.getString("price");
        String info = bud.getString("info");
        int cnt = bud.getInt("addshopcar");
        if (cnt > 0) {
            for (int i = 0; i < cnt; i++)
                shopcaritems.add(new Items(name, info, price));
            shopcarAdapter.notifyDataSetChanged();
        }
    }
}
```

7.商品详情页的信息显示。通过 Intent 从 MainActivity 拿到要显示的商品数据，然后将相应的变量赋值成对应的数据，其中商品图片比较笨，是通过一个一个判断是哪个商品来选图片。

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.itemdetail);
    intent=getIntent();//获取传递过来的消息
    name = intent.getStringExtra("Name");
    price = intent.getStringExtra("Price");
    info = intent.getStringExtra("Info");
    //商品信息
    Tname = (TextView)findViewById(R.id.itemdetail_r1_item_name);
    Tprice = (TextView)findViewById(R.id.itemdetail_r2_price);
    Tinfo = (TextView)findViewById(R.id.itemdetail_r2_info);
    img = (ImageView)findViewById(R.id.itemdetail_r1_picture);
    Tname.setText(name);
    Tprice.setText(price);
    Tinfo.setText(info);
    switch (name){
        case "Enchated Forest":
            img.setImageResource(R.drawable.enchatedforest);
            break;
        case "Arla Milk":
            img.setImageResource(R.drawable.arla);
            break;
        case "Devondale Milk":
            img.setImageResource(R.drawable.devondale);
            break;
        case "Kindle Oasis":
            img.setImageResource(R.drawable.kindle);
            break;
        case "waitrose 早餐麦片":
            img.setImageResource(R.drawable.waitrose);
            break;
        case "Mcvitie's 饼干":
            img.setImageResource(R.drawable.mcvitie);
            break;
        case "Ferrero Rocher":
            img.setImageResource(R.drawable.ferrero);
            break;
        case "Maltesers":
            img.setImageResource(R.drawable.maltesers);
            break;
        case "Lindt":
            img.setImageResource(R.drawable.lindt);
            break;
        case "Ferrero":
```

8.星星按钮的点击变化，就是点一次切换一次图片

```
//星星按钮变化
starbutton = (ImageButton) findViewById(R.id.itemdetail_r1_starbutton);
starbutton.setTag("0");
starbutton.setOnClickListener((view) -> {
        if (starbutton.getTag() == "0") {
            starbutton.setImageResource(R.drawable.full_star);
            starbutton.setTag("1");
        } else {
            starbutton.setImageResource(R.drawable.empty_star);
            starbutton.setTag("0");

        }
});
```

9.点击加入购物车按钮，通过一个变量 addshopcar 记录当前商品要加几个进去，到时作为 Intent 传回给父 activity。

```
//点击购物按钮
shopcarbutton= (ImageButton) findViewById(R.id.itemdetail_r2_shopcar);
shopcarbutton.setOnClickListener((view) -> {
        Toast.makeText(getApplicationContext(),"商品已添加到购物车",Toast.LENGTH_SHORT).show();
        addshopcar += 1;
});
```

10.点击返回按钮，将商品信息以及要增加几个进购物车的信息作为结果返回给父 activity。

```
//点击返回按钮
shopcarbutton= (ImageButton) findViewById(R.id.itemdetail_r1_backbutton);
shopcarbutton.setOnClickListener((view) -> {
        intent.putExtra("name",name);
        intent.putExtra("price",price);
        intent.putExtra("info",info);
        intent.putExtra("addshopcar",addshopcar);
        setResult(1,intent);
        finish();
});
```

11.取消界面顶部的标题栏，修改 styles.xml 使下图画红框部分一样即可

```
<!-- Base application theme. -->
<style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
    <!-- Customize your theme here. -->
    <item name="colorPrimary">@color/colorPrimary</item>
    <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
    <item name="colorAccent">@color/colorAccent</item>
</style>
```

（3） 实验遇到困难以及解决思路

1. 变量定义在子方法中时，需要加 final 关键字变成常亮，一开始不知道，报了很多错，后来我将会变的变量都作为 class 的全局变量而解决问题。

2. 不知如何在两个 activity 互相传递消息，后来上网查资料知道是使用 Intent 和 Bundle 来传。

3.之前实验用的是约束布局，这次主要使用 LinearLayout 和 RelativeLayout 来布局，由于之前没写过，一开始不知道如何下手，上网看了几个例子，并且通过动手尝试，终于知道如何使用 LinerLaryout 来做水平或者竖直的线性排列，以及用 RelativeLayout 来调整控件之间的位置关系。

4.关于详情页的图片，本来我是打算像 Name 那些一样可以放进数组传过去，不过由于图

片要用 setImageResource(R.drawable.xxx)来设置，一下子没找到如何将括号里的东西扔进数组，最后就通过直接 switch 一个一个判断这种暴力方法来解决。

五、 课后实验结果

本次实验的 ListView 的 Adapter 都用了自定义的 Adapter。不难，都是照着实验文档最后的拓展知识依样画瓢，最后

1. 购物车列表 ListView 的 Adapter，主要就是修改了 getView 和 ViewHolder,改为显示自己要显示的信息。

```
public class ShopcarAdapter extends BaseAdapter{
    private List<Items> items;
    private Context context;
    public ShopcarAdapter( List<Items> items, Context context){
        this.items=items;
        this.context = context;
    }
    @Override
    public int getCount() {
        if (items == null) {
            return 0;
        }
        return items.size();
    }
    @Override
    public Object getItem(int i) {
        if (items == null) {
            return null;
        } else {
            return items.get(i);
        }
    }
    @Override
    public long getItemId(int i) { return i; }
public View getView(int i, View view, ViewGroup viewGroup) {
    //新声明一个View变量和ViewHolder变量
    View convertView;
    ViewHolder holder;

    //当view为空时才加载布局，并且创建一个ViewHolder，获得布局中的3个控件
    if (view == null) {
        //通过inflate的方法加载布局，
        convertView= LayoutInflater.from(context).inflate(R.layout.shopcar, null);
        holder = new ViewHolder();
        holder.Firstletter=(TextView)convertView.findViewById(R.id.shopcar_icon);
        holder.Name = (TextView) convertView.findViewById(R.id.shopcar_name);
        holder.Price = (TextView) convertView.findViewById(R.id.shopcar_value);
        convertView.setTag(holder);//用setTag方法将处理好的viewHolder放入view中
    } else {//否则，让convertView等于view，然后从中取出ViewHolder即可
        convertView = view;
        holder = (ViewHolder) convertView.getTag();
    }
    //从viewHolder中取出对应的对象，然后赋值给它们
    holder.Firstletter.setText(items.get(i).getName().substring(0,1).toUpperCase());
    holder.Name.setText(items.get(i).getName());
    holder.Price.setText(items.get(i).getPrice());
    //将这个处理好的view返回
    return convertView;
}
private class ViewHolder {
    public TextView Name;
    public TextView Price;
    public TextView Firstletter;
}
}
```

2. 商品详情页底部功能栏的 Adapter

```java
public class OptionAdapter extends BaseAdapter{
    private Context context;
    List<String> options;
    public OptionAdapter(Context context, List<String> options) {
        this.context = context;
        this.options = options;
    }
    @Override
    public int getCount() {
        if (options != null) {
            return options.size();
        } else return 0;
    }
    @Override
    public Object getItem(int i) {
        if (options == null) {
            return null;
        } else {
            return options.get(i);
        }
    }
    @Override
    public long getItemId(int i) { return i; }
    @Override
    public View getView(int position, View view, ViewGroup viewGroup) {
        View convertView;
        ViewHolder holder;
        if (view == null) {
            convertView = LayoutInflater.from(context).inflate(R.layout.option, null);
            holder = new ViewHolder();
            holder.option = (TextView) convertView.findViewById(R.id.option);
            convertView.setTag(holder);
        } else {
            convertView = view;
            holder = (ViewHolder) convertView.getTag();
        }
        holder.option.setText(options.get(position));
        return convertView;
    }
    private class ViewHolder {
        public TextView option;
    }
}
```

3.生成功能列表

```java
//详情页底部功能列表
option =  new ArrayList<String>();
for (int i = 0; i < OptionName.length; i++)
    option.add(OptionName[i]);
mOptionAdapter = new OptionAdapter(ItemsDetails.this,option);
OptionView = (ListView) findViewById(R.id.itemdetail_option);
OptionView.setAdapter(mOptionAdapter);
```

六、 实验思考及感想

这次实验我觉得自己还是学到了很多知识，首先是学会了使用线性布局和关系布局。然后熟悉了 Java 类的定义以及一些接口的实现。知道了如何使用 Intent 和 Bundle 在 Activity 之间传递消息。对于 ListView 和 RecyclerView 的自定义适配器的写法和用法也有了了解。