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	15	专业（方向）	互联网
学号	15352006	姓名	蔡丽芝
电话	13538489980	Email	314749816@qq.com
开始日期	2017/10/23	完成日期	2017/10/24

一、 实验题目

Intent, Bundle 的使用以及 RecyclerView, ListView 的应用

二、 实现内容

本次实验模拟实现一个购物 app，主要有 3 个页面，分别是商品列表页，购物车页面，商品详情页面

实验要求：

实验布局要求：

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。

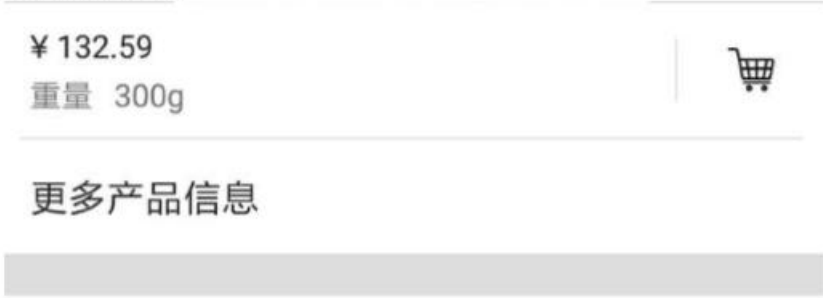
2、购物车列表界面

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。这一块建议大家使用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。

逻辑实现要求：

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。

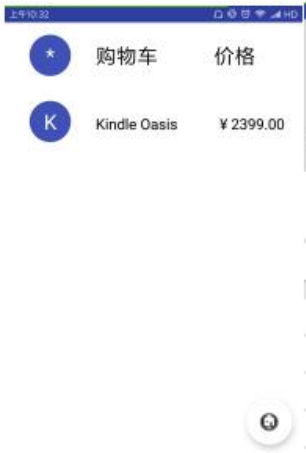
4、商品详情界面中点击返回图标会返回上一层，点击星标会切换状态，如果原先是空心星星，则会变成实心星星；如果原先是实心星星，则会变成空心星星。点击购物车图标会将该商品添加到购物车中并弹出Toast提示:"商品已添加到购物车"。

注:不要求判断购物车是否已有该商品,即如果已有一件该商品,添加之后则显示两个即可。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。

三、 课堂实验结果

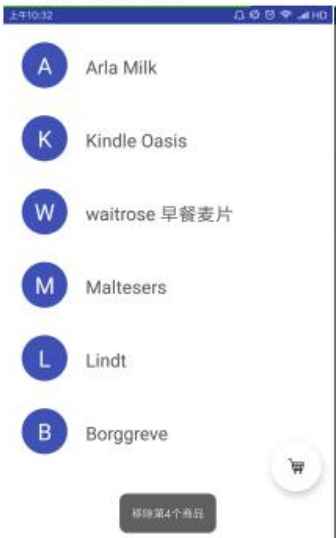
（1）实验截图

商品列表页

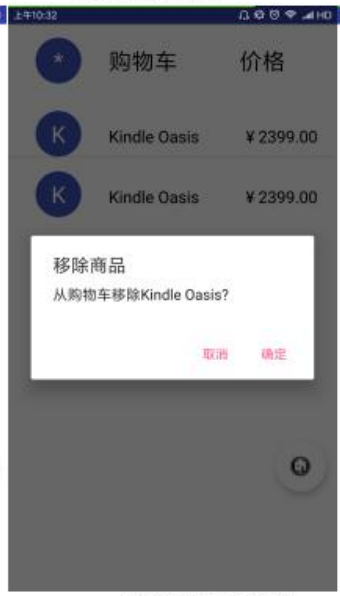


购物车列表页

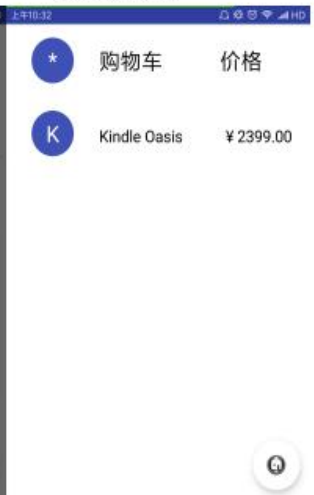
商品详情页



长按删除



长按删除购物车



长按删除后

（2）实验步骤以及关键代码

(1) Activity

```
<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=".ShowDetails"></activity>
```

创建两个 Activity 分别表示主页面即商品列表页和商品详情页

（2）主页面设置

```
<android.support.v7.widget.RecyclerView
    android:id="@+id/goods"
    android:background="#fff"
    android:layout_width="368dp"
    android:layout_height="551dp"
    tools:layout_editor_absoluteY="0dp"
    tools:layout_editor_absoluteX="8dp"
    />
```

主页面整体使用相对布局，将 RecyclerView 控件放入布局文件中，并为其设置固定唯一的 id 号 goods.

```
mRecyclerView = (RecyclerView) findViewById(R.id.goods);
mRecyclerView.setLayoutManager(new LinearLayoutManager(this));
mgoodsAdapter = new goodsAdapter(good, MainActivity.this);
mRecyclerView.setAdapter(mgoodsAdapter);
```

在 MainActivity.java 文件里通过 id 获取 RecyclerView，调用 setAdapter 使用自定义的适配器 goodsAdapter 为 RecyclerView 填充数据

自定义的 RecyclerView.Adapter, goodsAdapter

```
@Override
public MyViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
    View view = LayoutInflater.from(context).inflate(R.layout.gooditem,parent, false);
    MyViewHolder viewholder = new MyViewHolder(view);
    return viewholder;
}
```

创建 Item（自定义的商品 item 的）视图，并返回相应的 viewHolder,。

```
class MyViewHolder extends RecyclerView.ViewHolder{
    TextView tv;
    TextView tv2;
    public MyViewHolder(View view) {
        super(view);
        tv = (TextView)view.findViewById(R.id.circle);
        tv2 = (TextView)view.findViewById(R.id.good_id);
    }
}
```

、

通过 id 的方式从 View 中找到适配器所需的各个控件，最终形成一个完整的适配器

```
public int getItemCount() {
    return good.size();
}
```

告诉 RecyclerView-Adpater 列表 items 的总数，适配器在工作的时候，首先会检查需要适配的数据的大小，如果为 0，适配器就不会往 view 中适配，如果不为 0，就会根据大小往 view 中循环适配每一个数据

```
public void onBindViewHolder(final MyViewHolder holder, final int position) {
    holder.tv.setText(good.get(position).getName().substring(0,1).toUpperCase());
    holder.tv2.setText(good.get(position).getName());
    if(mOnItemClickListener != null) {
        holder.itemView.setOnClickListener((v) -> {
                mOnItemClickListener.onClick(position);
        });
        holder.itemView.setOnLongClickListener((v) -> {
                mOnItemClickListener.onLongClick(position);
                return false;
        });
    }
}
```

```
public interface OnItemClickListener {
    void onClick(int position);
    void onLongClick(int position);
}
```

```
public void setmOnItemClickListener(OnItemClickListener onItemClickListener) {
    this.mOnItemClickListener = onItemClickListener;
}
```

onBindViewHolder 函数将所需的数据绑定到正确的 Item 视图上，然而，因为 RecyclerView 本身没有点击触发事件，所以在这一部分需要在适配器中实现，首先在适配器中设置一个

监听器，当 itemView 点击的时候，调用该监听器并同时将 itemView 的 position 作为参数传递出去

```
<TextView
    android:id="@+id/circle"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="20dp"
    android:layout_marginTop="20dp"
    android:background="@drawable/circle"
    android:gravity="center_horizontal|center_vertical"
    android:text="e"
    android:textColor="#fff"
    android:textSize="25dp" />

<TextView
    android:id="@+id/good_id"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="20dp"
    android:layout_marginTop="20dp"
    android:gravity="center_vertical"
    android:text="Enchated Forest"
    android:textSize="20dp"
    android:textColor="#555"/>
```

```
good = new ArrayList<Goods>();
```

```
mgoodsAdapter = new goodsAdapter(good, MainActivity.this);
mRecyclerView.setAdapter(mgoodsAdapter);
```

对于 RecyclerView 这个容器中，由于我们是将每个商品的 item 模板适配到该容器中，所以可以首先在布局文件中自定义一个商品 item 模板，然后将这些 item 组合成 List 结构传递给适配器，适配器会自动把 List 的每一项与模板匹配起来

```
mgoodsAdapter.setmOnItemClickListener(new goodsAdapter.OnItemClickListener() {
    @Override
    public void onClick(int position) {
        Intent intent = new Intent(MainActivity.this, ShowDetails.class);
        intent.putExtra("Name", good.get(position).getName());
        intent.putExtra("Price", good.get(position).getPrice());
        intent.putExtra("Info", good.get(position).getInfo());
        startActivityForResult(intent,0);
    }
    @Override
    public void onLongClick(int position) {
        Toast.makeText(MainActivity.this,"移除第"+String.valueOf(position+1)+"个商品",Toast.LENGTH_SHORT).show();
        good.remove(position);
        mgoodsAdapter.notifyDataSetChanged();
    }
});
```

在 MainActivity.java 文件中，由于在 goodsAdapter.java 中我们已经给适配器中的每个 item 自定义了 setOnItemClickListener，所以这里可以添加点击事件，当只是点击商品的时候，会跳到商品详情页面，商品详情页面是另外一个 activity，所以第一个红方框中首先创建需要启动的 Activity 即 showDetails 的 Intent，然后通过 putExtra 用 key-value 的方式将对应的数据传递到另外一个 activity 中，当长时间点击事件的时候，首先会调用 Toast 提示第几个商品被删除，然后直接从 good 中将第 position 位移除就可以了，最后调用 notifyDayaSetChanged 函数更新整个 ReyclerView。

(3) 购物车列表页

为了使得逻辑更加明确，我将购物车列表页和商品列表页放在了同一个布局文件中，然而由于每次只能显示一个页面，所以首先调用 setVisibility 函数将购物车列表页设为 invisible

```
final LinearLayout shopCartView = (LinearLayout) findViewById(R.id.shopList);
shopCartView.setVisibility(View.INVISIBLE);
```

```
final ImageButton change = (ImageButton) findViewById(R.id.car);
change.setTag("0");
change.setOnClickListener((view) -> {
        if (change.getTag() == "0") {
            change.setImageResource(R.drawable.mainpage);
            change.setTag("1");
            shopCartView.setVisibility(View.VISIBLE);
            mRecyclerView.setVisibility(View.INVISIBLE);
        } else {
            change.setImageResource(R.drawable.shoplist);
            change.setTag("0");
            shopCartView.setVisibility(View.INVISIBLE);
            mRecyclerView.setVisibility(View.VISIBLE);
        }
});
```

给 change 设置 tab，通过 tag 来判断当前 view 的状态，初始化 view 的 tag 为 0，如果点击 change 按钮的时候，view 的 tag 为 0，表明当前是商品列表，需要切换到购物车列表，并将 tag 置为 1；如果点击 change 按钮的时候，view 的 tag 为 1，那么切换到商品列表，并将 tag 置为 0

```
mListView = (ListView) findViewById(R.id.shopCarItem);
mListView.setAdapter(mshopCarAdapter);
mListView.setOnItemClickListener((adapterView, view, i, l) -> {
        Intent intent = new Intent(MainActivity.this, ShowDetails.class);
        intent.putExtra("Name", shopCarGood.get(i).getName());
        intent.putExtra("Price", shopCarGood.get(i).getPrice());
        intent.putExtra("Info", shopCarGood.get(i).getInfo());
        startActivityForResult(intent,0);
});
mListView.setOnItemLongClickListener((adapterView, view, i, l) -> {
        final AlertDialog.Builder alertDialog = new AlertDialog.Builder(MainActivity.this);
        alertDialog.setTitle("移除商品").setMessage("从购物车移除" + shopCarGood.get(i).getName()+ "?")
                .setPositiveButton("确定", (dialogInterface, which) -> {
                        if(shopCarGood.remove(i) != null) {
                            mshopCarAdapter.notifyDataSetChanged();
                        }
                }).setNegativeButton("取消", (dialog, which) -> {
                Toast.makeText(getApplicationContext(), "你选择了取消", Toast.LENGTH_SHORT).show();
        }).show();
        return true;
});
```

当点击的时候，同商品列表页通过 intent 启动一个新的 activity，并通过 putExtra 的方式传递信息，当长点击的时候，设置一个对话框，如果按确定，就删除该物品，并调用 notifyDataSetChanged 更新整个 RecyclerView

（4）商品详情页

```
final String showName = intent.getStringExtra("Name");
final String showPrice = intent.getStringExtra("Price");
final String showInfo = intent.getStringExtra("Info");
```

在点击商品的时候，有通过 putExtra 的方式传递消息，在 ShowDetails.java 中通过 getStringExtra(key)的方式获得对应的值

```
class MyViewHolder extends RecyclerView.ViewHolder {
    TextView tv;
    TextView tv2;
    TextView tv3;
    ImageView pc;
    ImageButton back;
    ImageButton star;
    ImageButton addgoods;
    ListView glist;
    public MyViewHolder(View view) {
        super(view);
        tv = (TextView)view.findViewById(R.id.name);
        tv2 = (TextView)view.findViewById(R.id.price);
        tv3 = (TextView)view.findViewById(R.id.weight);
        pc = (ImageView)view.findViewById(R.id.pic);
        back = (ImageButton)view.findViewById(R.id.back);
        star = (ImageButton)view.findViewById(R.id.star);
        addgoods = (ImageButton)view.findViewById(R.id.addShopCar);
        glist = (ListView)view.findViewById(R.id.function);
        star.setTag("0");
    }
}
```

自定义一个新的 Viewholder，从 view 中找到各个控件， 以便绑定新的内容

```
holder.tv.setText(Name);
holder.tv2.setText(Price);
holder.tv3.setText(Info);
```

```
holder.glist.setAdapter(fun);
```

```
holder.back.setOnClickListener((view) -> { act.finish(); });

holder.star.setOnClickListener((view) -> {
        Object tag = holder.star.getTag();
        if(tag == "0") {
            holder.star.setTag("1");
            holder.star.setImageResource(R.drawable.full_star);
        } else {
            holder.star.setTag("0");
            holder.star.setImageResource(R.drawable.empty_star);
        }
});

holder.addgoods.setOnClickListener((view) -> {
        Toast.makeText(context, "商品已添加到购物车", Toast.LENGTH_SHORT).show();
        cnt++;
        intent.putExtra("cnt", cnt);
        intent.putExtra("name", Name);
        intent.putExtra("Price", Price);
        intent.putExtra("info", Info);
        act.setResult(0,intent);
});
```

通过 holder 将布局模板中的每个控件赋值，其中 back 代表后退图标，act 表示当前的 activity，当点击回退图标的时候，结束当前的 activity，返回上一层 activity。同购物车按钮一样，给 view 设置一个 tag，通过 tag 判断当前状态，如果点击时，tag 为 0，就将 tag 置为 1，并改变图片资源。当点击购物车按钮的时候，通过 putExtra 和 setResult 将添加商品到购物车的行为信息返回给 MainActivity.

```
protected void onActivityResult(int requestCode, int resultCode, Intent intent) {
    if(requestCode == 0 && resultCode == 0) {
        Bundle bud = intent.getExtras();
        String str = bud.getString("name");
        String pri = bud.getString("Price");
        String info = bud.getString("info");
        int cnt = bud.getInt("cnt", 0);
        for(int i = 0; i < cnt; i++) {
            shopCarGood.add(new Goods(str, info, pri));
        }
        mshopCarAdapter.notifyDataSetChanged();

    }
}
```

在 MainActivity 中，通过重写 onActivityResult 函数，利用 Bundle 获得商品详情页面返回的信息，然后将新增的商品添加到 cargood 中，并调用 notifyDataSetChanged 进行更新。

（3）实验遇到困难以及解决思路

答：实验过程一开始对于 intent 和 bundle 的使用不了解，不知道如何在两个不同的 activity 中进行数据的传递，后来通过查阅相关的资料，了解到可以使用 putExtrade 和 getStringExtra 和 Bundled 的方式进行数据的传递。其次就是遇到点击商品要跳转到商品详情页面的时候，app 会出现闪退的情况，后来通过百度知道，原来是没有在 AndroidManifest.xml 中进行注册新的 activity

四、 实验思考及感想

这次实验总的来说，对我是一个巨大的挑战，一开始真的不知道如何下手，不知道如何布局，不知道如何使用 RecyclerView 和 ListView, 但是最后通过百度和查阅官方文档，终于写完了这次 project，还是有点小成就的，希望自己下次实验，继续努力